Finite Element and Neural Network Solvers for Modelling Microcirculation

Martin Kristiansen

Thesis submitted for the degree of
Master in Computational Science: Physics
60 credits

Department of Physics
Department of Mathematics
Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Spring 2023

Finite Element and Neural Network Solvers for Modelling Microcirculation

Martin Kristiansen

Finite Element and Neural Network Solvers for Modelling Microcirculation

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Abstract

Through densely connected networks of vessels, microcirculatory systems exchange blood in the vasculature and the surrounding tissue. For finite element solvers, these vast microcirculatory systems can offer challenges in preprocessing and/or inference. This motivates the study of supplementary ways of obtaining the relation between vascular geometry and fluid interaction, and our objective is to see if the Fourier neural operator can learn this relation when we model fluid interaction by diffusion and exchange in branching geometries continuously connected by one-dimensional segments and in the confining two-dimensional square.

Since our proposed model learns in the supervised setting we first derive the method of constrained constructive optimization (CCO), allowing us to algorithmically create realistic vascular trees. In this way, we circumvent the need for medical images and can decide the complexity of the vascular geometry by deciding the number of vessels in the vascular tree. With coupled partial differential equations defined on 2D-1D manifolds we describe fluid interaction and by the conditions of the Lax-Milgram theorem, we address the existence and uniqueness of weak solutions. By regularity assumptions, we develop a priori error estimates and investigate convergence rates of the finite element solution to manufactured solutions on different 2D-1D manifolds.

Having established confidence in the finite element solver we develop a data production routine to obtain the input by a distance representation of the vascular trees and target by the finite element solver. We train and test the machine learning model through a series of experiments and obtain 3% relative mean squared test error for 10000 (20% test, 80% train) samples of vascular trees with 3-5 vessels. Best and worst predictions are visualized and interpreted and by increasing the number of vessels we look at the generalization of the proposed model architecture to the higher complexity of the vascular trees. Finally, we verify the property of discretization independence of the model and illustrate the time-sparing advantage it can offer compared to the finite element solver.

Acknowledgements

I want to thank my supervisors Miroslav Kuchta and Kent-Andre Mardal, this thesis could never happen without your ideas and support. I also want to thank the master's programme of Computational Science for opening up the barriers between physics, mathematics and informatics at UiO. Lastly, I want to thank the Simula Research Laboratory for setting me on the path to this awesome multidisciplinary thesis.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

The blood circulation in the human brain and other organs is through vast vascular networks of arteries, arterioles, capillaries, venules and veins. The arteries, carrying oxygenated and nutrient-rich blood from the heart, branch out to arterioles which supply networks of capillaries, known as capillary beds. Capillaries connect arterioles and veins and are responsible for exchanging oxygen and other nutrients as well as clearance of waste substances like carbon dioxide. Understanding solute transport in such microcirculatory systems is of great physiological importance and can contribute to advances in e.g. drug delivery (Cattaneo & Zunino, 2014) or a better understanding of Alzheimer's disease (Goirand et al., 2021). There are three distinct ways of studying solute transport in microvasculature: *in vivo*, *in vitro* and *in silico*. In microbiology, in vivo refers to conducting a study in living organisms. In vivo studies are often based on different imaging techniques, e.g. magnetic resonance imaging (MRI) and photoacoustic imaging. Through images, one can obtain information on the spatial arrangement of the vasculature, morphometric properties and also measures of function by images of perfusion, oxygenation or permeability (A. Murray & Dinsdale, 2016). In vitro refers to the study of isolated components of the organism. Because of the complex network of vessels in the microvasculature, in vitro studies can provide more accurate measurements by isolating components of this vast connected network of vessels (see Kodama et al., 2019, e.g.). In silico refers to studies conducted on computers and via computer simulations. There are numerous ways of studying microcirculatory systems in silico and these studies can also be combined with in vivo measurements. Often in vivo image data is used to obtain 3D models of microvascular networks and by prescribing fluid equations to the vascular system, one can study the solute transport and delivery to the surrounding tissue (see e.g. Linninger et al., 2013; Peyrounette et al., 2018). In this thesis, we will conduct our study purely in silico, and to obtain numerical methods we must first obtain a mathematical model of fluid interaction. The proposed model is based on the seminal work of (D'Angelo, 2007) which through dimensionality reduction of the vessels modelled networks of vessels and their interaction with the surrounding tissue. Dimensionality reduction offers relief in the computational expense while retaining geometric properties of the vasculature and is also a well-investigated mathematical technique for studying microcirculatory systems (see e.g. D'Angelo, 2012; Kuchta et al., 2016; Masri et al., 2023). However, for increasing complexity in the vasculature and to reduce the approximation error, these finite element models increase in computation expense in preprocessing

and/or inference. For body-fitted meshes that require conformity between the discretization of the tissue and that of the vessels, the computational cost increases in preprocessing, i.e. in obtaining the mesh needed for the finite element solver. Furthermore, (D'Angelo, 2012) showed that to retain optimal convergence rates the refinement of the mesh had to be increased in the vicinity of the vasculature. To by-pass this need for high refinement alternative methods have been proposed (see e.g Gjerde et al., 2020; Koch et al., 2020) but these alternatives again rely on higher complexity in inference, to assemble and solve the corresponding linear system. Because of this, it is interesting to study alternative methods that can offer time-sparing advantages both in preprocessing and inference. Such methods exist via artificial neural networks (NN's) and the use of NN's to assist or improve computational fluid dynamics (CFD) calculations has become very popular in recent years (see figure 1. Viquerat & Hachem, 2020) but examples of use in biomedical research related to microcirculatory flow are sparse and primarily by the use of physics-informed neural networks (PINN's) (Karniadakis et al., 2002).

While restricting ourselves to the two-dimensional setting our objective is to investigate if NN's and specifically if the Fourier neural operator (FNO) (Li et al., 2020a) can offer a mesh-free and discretization-independent alternative way of obtaining information on the interaction of fluid in microvascular systems and the surrounding tissue.

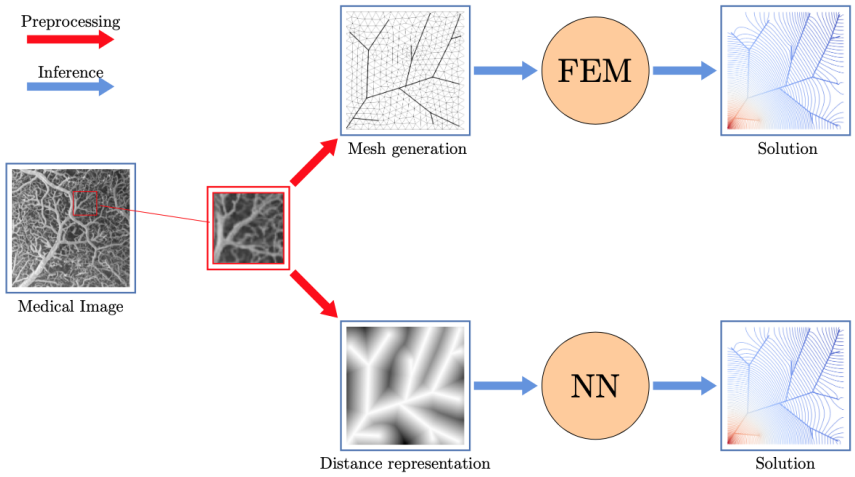


Figure 1.1: Schematic display of obtaining pressure distributions. 1) By mesh generation and the finite element method. 2) By distance representation and a pre-trained neural network. The medical image is obtained from (Meyer et al., 2008) and is an image of vascular corrosion cast at $400\mu m$ of the cerebral cortex of mice. The structures in the other images are made to resemble an arbitrary piece of an artery and are obtained using the method of constrained constructive optimization.

Figure 1.1 illustrates how trained neural networks can offer an alternative to the traditional finite element approach in obtaining the solution corresponding to the pressure distribution in the vasculature and the surrounding tissue. Since we aim to develop an alternative path through supervised learning we will

in chapter 2 first construct a way of obtaining realistic vascular trees using the method of constrained constructive optimization (CCO). This will remove the dependence on the use of medical images to obtain vascular geometries with the additional benefit of being able to control the geometric complexity of the vasculature, by deciding the number of vessels in the vascular trees. In chapter 3 we derive a mathematical model of the interaction of fluid in microvessels and the surrounding tissue. By model reduction of the vessels, going from a two-dimensional representation to a one-dimensional representation we can easily extend the model to hold for vascular trees. Fixing appropriate boundary conditions and including the method of CCO, we obtain a purely numerical method, in the framework of the finite element method, to investigate how different vascular geometries correspond to different pressure distributions along the vasculature and in the confining square representing the tissue. This represents the first path in Figure 1.1 and is the way we will generate the data needed for the Fourier neural operator. Chapter 4 describes the Fourier neural operator, its architecture and parameter choices and explains how we choose to represent the vascular trees as input and how we represent the pressure in the vasculature and the surrounding tissue as the target. In chapter 5 and with a simplified dataset of input-target samples, we train and reconfigure according to a parameter search before we step by step increase the complexity of the dataset and the FNO architecture to obtain our final configuration of the neural network model. We measure how the Fourier neural operator generalizes to higher complexity of the vasculature, its discretization independence and execution time compared to that of obtaining the pressure from the finite element method which we used to generate the data.

Appendices

Appendix A contains additional experiments on the harmonic extension of the pressure distribution in the vascular geometry to the confining square square. The experiments ensure the correct implementation of the interior boundary conditions and also provides convergence analysis of the harmonic extension to the pressure distribution in the confining square.

Appendix B contains a simplified, less realistic approach to obtaining branching structures continuously connected by one-dimensional segments by the method of rapidly exploring random trees.

Mathematical notation

a lower case letters are used for scalars and scalar functions.

A upper case letters are used for matrices, operators, functionals and spaces, but also to separate scalar values.

$\mathbf{a}$ bold letters are used for points, vectors and vector-valued functions.

$\mathcal{A}$ calligraphic letters are used for sets.

Lastly, we use $\langle \cdot, \cdot \rangle$ to denote the inner product and use the notation $[\cdot]$ and $(\cdot)$ interchangeably.

CHAPTER 2

Constrained Constructive Optimization

Since the microcirculatory systems consist of vast networks of microvessels, the possibility to construct rather than extract such networks can offer time-sparing advantages, as well as circumvent the lack of information about vascular architecture. Therefore, numerous efforts have been made to develop automatic generation of vascular networks and there are especially two classes of methods that are popular in the literature. 1) Fractal methods (see e.g. Barnsley & Demko, 1985; Bassingthwaighte et al., 1989; Gottlieb, 1990; West & Goldberger, 1987) and 2) Space filling methods (see e.g. Nekka et al., 1996; Schreiner & Buxbaum, 1993). Fractal methods follow the statistical evaluation of features like vessel radii, vessel length, bifurcation, etc. and unlike the space-filling methods, the fractal methods do not provide confinement on the vascular network but are independent of the vascular territory. In our approach, we will follow (Schreiner & Buxbaum, 1993) deriving the space-filling method of constrained constructive optimization and although the method is based on physical laws rather than statistical observation. One key feature of the method is also statistically observed. The feature is the relation of the radii through bifurcations and is known as Murray's law (C. D. Murray, 1926) observed in mammalian circulatory and respiratory systems (Sherman, 1981).

(Schreiner & Buxbaum, 1993) introduced the method of constrained constructive optimization (CCO) to generate arterial tree structures by modelling the arterial trees as networks of straight cylindrical tubes following Poiseuville's law (Sutera & Skalak, 1993). Furthermore, the straight cylindrical tubes representing the vessels in the arterial tree were modelled with a boundary condition, in the form that each terminal vessel should output the constant flow rate Q_{term} against the constant pressure P_{term}. The task of the generated tree is then to supply the tissue with the prescribed flow Q_0 while following a set of geometrical constraints. They found that the requirements above could be fulfilled by appropriate scaling of the vessels' radii and independent of the particular connective structure of the tree. By using an appropriate cost function dependent on the segments' radii they were able to connect terminal vessels to a preexisting tree simply by choosing the connective structure which minimizes the cost. The method was further developed by the original authors in (Karch et al., 1999, 2000) and by other authors (see e.g Jaquet et al., 2018; Maso Talou et al., 2021).

2.1 The method of CCO

Bifurcation rule

Considering a parent vessel v_p bifurcating into children vessels v_n, v_{new}, we constrain the vessels' radii before and after bifurcation to follow Murray's law

$$r_{v_p}^{\gamma} = r_{v_n}^{\gamma} + r_{v_{new}}^{\gamma} \tag{2.1}$$

where γ is called the bifurcation exponent. The specific choice of γ has been discussed in e.g. (Brown et al., 2000) and varies between 2 and 3, but it is most commonly chosen as $\gamma = 3$ following the original derivation in (C. D. Murray, 1926) from Poiseuville's law.

Cost function

Consider a vessel v_i uniquely defined by $(r_i, \mathbf{x}_i^p, \mathbf{x}_i^d)$, where r_i is the vessel radius, $\mathbf{x}_i^p$ its start/proximal point and $\mathbf{x}_i^d$ its end/distal point, the length of the vessel is then $l_i = \|\mathbf{x}_i^d - \mathbf{x}_i^p\|$ and its two dimensional volume is given by $V_i = l_i \pi r_i^2$. Let $\mathcal{T}$ denote a generated tree consisting of the set of vessels $\{v_i, i = 1, .., N\}$. By defining the cost function equivalent to the total volume of $\mathcal{T}$ we aim to minimize

$$V(\mathcal{T}) = \sum_{i=1}^{N} l_i \pi r_i^2. \tag{2.2}$$

Hemodynamics model

If we presume the blood to behave as an incompressible, homogeneous Newtonian fluid and let each vessel be a straight cylindrical tube we can apply Poiseuille's law to find the resistance in each vessel v_i by

$$R_i = \frac{8\eta}{\pi} \frac{l_i}{r_i^4} \tag{2.3}$$

where η is the viscosity of the blood. Additionally, we have

$$\Delta p_i = R_i Q_i \tag{2.4}$$

where Δp denotes the pressure drop in v_i and Q_i the volumetric flow rate. We note the following properties of the resistance for later use; Considering vessels in series we have additivity of the resistance

$$R_{tot} = R_1 + R_2 \tag{2.5}$$

and in parallel, we have the relation

$$\frac{1}{R_{tot}} = \frac{1}{R_1} + \frac{1}{R_2}. \tag{2.6}$$

2.2 Implementation

We initialize our method by defining a root vessel v_0 inside or partially inside the *perfusion area* Ω_p with proximal point $\mathbf{x}_0$ and distal point $\mathbf{x}_1$. We then

draw a random number $\mathbf{x}_d$ inside Ω_p and seek to connect the random number $\mathbf{x}_d$ to our tree $\mathcal{T}$, which at this point only consists of the root. The connection will then replace v_0 with tree new vessels v_p, v_n, v_{new} where $v_p = (\_, \mathbf{x}_0, \mathbf{x}_{b_k})$ is a straight vessel, bifurcating into $v_n = (\_, \mathbf{x}_{b_k}, \mathbf{x}_n)$ and $v_{new} = (\_, \mathbf{x}_{b_k}, \mathbf{x}_{new})$ such that $\mathbf{x}_d = \mathbf{x}_{new}$ is the distal end of v_{new} and $\mathbf{x}_n = x_1$ - the distal end of the root becomes the distal end of v_n. We denote $\mathcal{T}_k = (v_p, v_n, v_{new})_k$ the proposed resulting tree with specific bifurcation $\mathbf{x}_{b_k}$. The *optimal bifurcation point* $\mathbf{x}_b$ is then chosen such that we minimize the cost function (2.2). Note, that the radii in v_p, v_n and v_{new} are omitted since this depends on the exact location of the bifurcation candidate x_{b_k} and thus needs to be calculated for all candidate bifurcation points $\mathbf{x}_{b_k}$ to obtain the different costs $V(\mathcal{T}_k)$. This is achieved by scaling the vessels' radii such that the terminal vessels yield the constant flow Q_{term} against the constant pressure p_{term}. Presuming we have a tree $\mathcal{T} = \{v_i, i = 1, .., N\}$ of N connected vessels we proceed as follows.

Perfusion area

The perfusion area denotes the area for which the new terminal vessel is allowed to output the constant terminal flow Q_{term}. Because the main structure of the tree $\mathcal{T}$ is highly influenced by the first vessels we want to discard distal points to our new terminal vessel close to the existing tree. This is achieved by defining a minimum distance l_{min} such that the randomly drawn point $\mathbf{x}_d$, following a uniform distribution inside the perfusion area Ω_p, fulfills

$$\|\mathbf{x} - \mathbf{x}_d\| > l_{min} = l_c \sqrt{\frac{\nu}{N_{term} + 1}}. \tag{2.7}$$

$\mathbf{x}$ is here the closest point in the set of vessels in the preexisting tree $\mathcal{T}$, $l_c = \sqrt{\frac{\int_\Omega dA}{\pi}}$, N_T is the number of terminal vessels in $\mathcal{T}$ and ν is a tuning parameter. Should N_{fail} consecutive attempts to draw a random point fulfilling (2.7), we let

$$l_{min} = f_r l_{lim} \tag{2.8}$$

for $f_r \in (0, 1)$. The possible connection points in the tree $\mathcal{T}$ are the proximal points in each vessel except the root. Additionally, we consider a fixed number N_{con} of vessels to be evaluated for connecting the randomly drawn point $\mathbf{x}_d$. Once we obtain a point $\mathbf{x}^d$ fulfilling (2.7), we let $\mathbf{x}_{new} = \mathbf{x}^d$ indicating that this will be the distal point in our new terminal vessel. The candidate vessels (to be replaced) are denoted $\mathcal{T}_n = \{v_i, .., v_{N_{con}}\}$ and are the set of N_{con} vessels closest to the new terminal point $\mathbf{x}_{new}$. Note that if the preexisting tree has a number of vessels less than N_{con}, then $\mathcal{T} = \mathcal{T}_n$.

Flow rate

Since the addition of a new terminal segment increases the flow rate in our tree we need to update the flow rate in each vessel accordingly. If we let each terminal segment supply the "tissue" Ω_p with the constant flow rate Q_{term}, the flow rate Q_i in each vessel v_i is uniquely defined by the number of terminal segments distal to v_i. Letting n^i_{dist} denote the number of terminal vessels distal to v_i we have

$$Q_i = n^i_{dist} Q_{term} \tag{2.9}$$

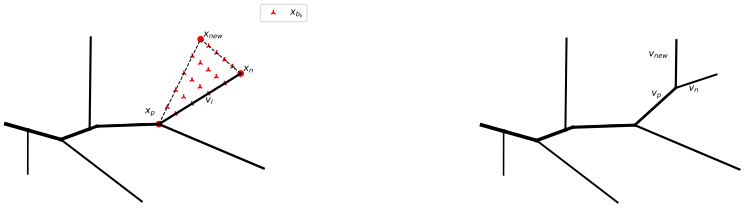
and thus, the input flow rate Q_0 is given by

$$Q_0 = N_{term} Q_{term} \tag{2.10}$$

where N_{term} is the current total number of terminal vessels in $\mathcal{T}$.

Bifurcation candidates

Once we have obtained the set of closest vessels $\mathcal{T}_n = \{v_i, .., v_{N_{con}}\}$, we discretize the triangle domain with vertices $(\mathbf{x}_i^p, \mathbf{x}_i^n, x_{new})$ into a set of candidate bifurcation points $\mathcal{X}_b = \{\mathbf{x}_{b_k}\}_{k=1}^{P_{opt}}$. $P_{opt} = \sum_{i=1}^{\Delta v} i - 3$ and Δv is the discretization of each side length in the triangle. We then seek to minimize the cost function



(a) Bifurcation point candidates $\mathcal{X}_b$

(b) Vascular tree $\mathcal{T}$ with optimal bifurcation point x_b

Figure 2.1: Illustration of optimal bifurcation point search, replacing vessel v_i with vessels (v_p, v_n, v_{new}) using $\Delta v = 6$.

(2.2) over all candidate bifurcations $(v_i^p, v_i^n, v_{new})_k$ replacing vessel $v_i \in \mathcal{T}_n$. We denote $\mathcal{T}_k^i$ the preexisting tree $\mathcal{T}$ with the specific vessel $v_i \in \{v_i\}_{i=1}^{N_{con}}$ replaced by $(v_p, v_n, v_{new})_k$ and with specific bifurcation $\mathbf{x}_{b_k} \in \mathcal{X}_b$. To obtain the radii needed to calculate the cost of the candidate tree $\mathcal{T}_k^i$ we proceed as follows.

Reduced resistance

Following the derivations made in (Karch et al., 1999) we define the reduced resistance $R^* = Rr^4$ and consider a possible set of candidate vessels (v_p, v_n, v_{new}) where v_n may or may not be terminal. For terminal segments, the reduced resistance is simply given by

$$R_{term}^* = \frac{8\eta}{\pi} l_i \tag{2.11}$$

while for a combined set of vessel, e.g. v_p bifurcating into v_n and v_{new} the resistance in v_p is given by

$$R_{sub,p} = \frac{8\eta}{\pi} l_i + \left[\frac{(r_n/r_p)^4}{R_n^*} + \frac{(r_{new}/r_p)^4}{R_{new}^*} \right]^{-1} \tag{2.12}$$

where we have used the relations (2.5) and (2.6). In the case v_n is not terminal, we calculate the resistance to v_n by post-order traversal. Starting at the

terminal segments distal to v_n and traversing up to v_n applying (2.11) at each terminal vessel and (2.12) at each non-terminal vessel distal to v_n. Notice for the situation when v_n is not terminal, the ratios (r_n/r_p) and (r_{new}/r_p) have been calculated in the preceding steps in the tree generation.

Bifurcation ratios

Exploiting the fact that we have equal terminal pressures and thus a constant pressure drop Δp between the root and an arbitrary terminal vessel, we have by (2.4)

$$R_n Q_n = R_{new} Q_{new} \rightarrow \frac{Q_n}{Q_{new}} = \frac{R_{new}}{R_n}$$

and using the factorization $R^* = Rr^4$ we obtain

$$\frac{r_n}{r_{new}} = \left(\frac{Q_n R_n^*}{Q_{new} R_{new}^*} \right)^{1/4}. \tag{2.13}$$

By the bifurcation rule (2.1) we can then obtain the bifurcation ratio

$$\left(\frac{r_p}{r_n} \right)^{\gamma} = 1 + \frac{r_{new}}{r_n}^{\gamma} \rightarrow \qquad \frac{r_p}{r_n} = \left[1 + \left(\frac{r_{new}}{r_n} \right)^{\gamma} \right]^{1/\gamma} \tag{2.14}$$

and similarly

$$\frac{r_p}{r_{new}} = \left[1 + \left(\frac{r_{new}}{r_n} \right)^{-\gamma} \right]^{1/\gamma}. \tag{2.15}$$

Knowing the bifurcation ratios given by the formulas above, we can obtain the radii in each vessel v_i by

$$r_i = r_0 \cdot \prod_{k=1}^{i} \beta_k, \tag{2.16}$$

where the product is taken over the unique path from the root to the vessel v_i and β_k are the bifurcation ratios along this path. The root radius r_0 is given by

$$r_0 = \left[R_0^* \cdot \frac{Q_0}{p_0 - p_{term}} \right]^{1/4}. \tag{2.17}$$

Since R_0^* obviously varies when growing the vascular tree i.e. adding more terminal vessels and Q_0 is fixed by (2.10) we have the option to either vary the root radii r_0 or the total pressure drop $\Delta p = p_0 - p_{term}$ to fulfil (2.17).

Update tree

We can now obtain the updated resistance in the unique path of vessels proximal to our new bifurcation site. Since clearly, the bifurcation ratios depend on the flow rates and the reduced resistances by (2.13), we also need to update all bifurcation ratios in the unique path proximal to v_p and then update the radii in the tree according to (2.16). Finally, we can obtain the cost $V(\mathcal{T}_k^i)$ for a specific vessel v_i (to be replaced) and a specific bifurcation point $\mathbf{x}_{b_k}$.

Geometric constraints

Along with the bifurcation rule (2.1) we include the following geometrical constraints.

- We do not allow for vessels to intersect.

When obtaining the set of candidate bifurcation points $\mathcal{X}_b$ we first check if the bifurcation point to be evaluated leads to intersecting vessels. If not, we proceed to calculate the cost. To speed up computation we temporary create a vessel $v_{temp} = (\_, \mathbf{x}_i^p, \mathbf{x}_{new})$ between the proximal point of $v_i \in \mathcal{T}_n$ and the new terminal point $\mathbf{x}_{new}$ for each $v_i \in \mathcal{T}_n$ and check if v_{temp} intersects with the other vessels in $\mathcal{T}_n$. If v_{temp} leads to an intersection we neglect the vessel v_i corresponding to v_{temp} in $\mathcal{T}_n$.

Following (Maso Talou et al., 2021) we require two more geometrical constraints.

- Symmetry ratio constraint

For terminal vessels, we require that the ratio between the minimum radius and the maximum radius, when considering two post-bifurcating vessels, to be bounded below by the symmetry ratio parameter δ

$$\frac{\min(r_n, r_{new})}{\max(r_n, r_{new})} > \delta. \tag{2.18}$$

This has the effect of a more homogeneous distribution of the terminal vessels.

- Aspect ratio constraint

The aspect ratio of each vessel v_i in the arterial tree $\mathcal{T}$ is constrained to follow

$$\frac{l_i}{r_i} > 2 \tag{2.19}$$

which bounds the length of each vessel to be at least two times the length of the radius.

2.3 CCO parameters

Table 2.1 summarize the parameters introduced in the preceding sections. The geometric parameters describe the geometric features of the model by specifying the confinement of the arterial tree (perfusion area) the initial root vessel and the total number of terminal vessels in the tree. The geometric parameters γ, δ ensure that geometric features of real vasculature are embedded in the model. The physical parameters can be set to real physiological values, representing specific parts of the vasculature, as they did in the original paper (Schreiner & Buxbaum, 1993) modelling the left anterior descending artery during cardiac arrest. Alternatively, the root radius can be fixed, then by (2.17) the value Δp does not enter the model other than requiring equal terminal pressures to obtain (2.14) and (2.15) and the flow rate Q only enters the model as a ratio between the terminal flow rate and the flow rate in the connecting vessel v_n, in addition to our requirement under **Flow rate**. This means we obey the

Geometric	Ω_p	perfusion area
	$\mathbf{x}_0, \mathbf{x}_1$	proximal and distal end of root vessel
	N^t_{final}	number of terminal vessels in the final tree $\mathcal{T}$
	γ	bifurcation exponent
	δ	symmetry ratio parameter
Physical	η	blood viscosity
	Q_0, Q_{term}	total perfusion flow rate or terminal flow rate
	r_0 or Δp	root radius or total perfusion pressure drop by (14)
Tuning	ν	tuning parameter
	f_r	correction step factor
Optimization	N_{fail}	correction number
	N_{con}	number of vessels evaluated in connection search
	Δv	discretization parameter for bifurcation point search

Table 2.1: Parameter specifications for Algorithm 1.

rules of the CCO but the value of Δp is not required and the flow rate Q_0 *or* Q_{term} can be set arbitrarily. Since the viscosity is constant for all vessels in the vascular tree, this can also be set arbitrarily for fixed r_0. The optimization parameters decide the attempts to fulfil (2.7) without the correction (2.8) and the number of cost evaluations (2.2) deciding each vessel connection. Increasing the optimization parameters thus results in a more predictable vascular growth but at increasing computational cost.

We choose to initialize the algorithm with *one* root vessel, but the algorithm can be grown from an arbitrary number of connected preexisting vessels following the prescribed rules.

2.4 CCO algorithm

We summarize the method described in chapter 2 with the following algorithm.

Algorithm 1 CCO

Data: Parameters as in Table 2.1

$n = 0$
while $n < N^t_{final} - 1$ **do**
  $j = 0$
  repeat ▷ (2.7) → (2.8)
    draw random point $\mathbf{x}^d$ inside Ω
    $j = j + 1$
    if $j = N_{fail}$ **then**
      $l_{min} = f_r l_{min}$
      $j = 0$
    end if
  until $\|\mathbf{x}^d - \mathbf{x}\| > l_{min}$
  Define $\mathcal{T}_n$
  for $v_i \in \mathcal{T}_n$ **do**
    Update flow ▷ (2.9)
    Compute bifurcation candidates $\mathcal{X}_b$
    for $\mathbf{x}_b \in \mathcal{X}_b$ **do**
      Update reduced resistance ▷ 2.11 → (2.12)
      Update bifurcation ratios ▷ (2.13) → (2.17)
      Update tree $\mathcal{T}_k$
      if Constraints is satisfied **then** ▷ (2.18) → (2.19)
        $\mathcal{C}_k = \mathcal{C}_k \cup V(\mathcal{T}_k)$
      end if
    end for
    if $\mathcal{C}_k \neq \emptyset$ **then**
      $\mathcal{C}_{v_i} = \mathcal{C}_{v_i} \cup \min_k(\mathcal{C}_k)$
    end if
  end for
  if $\mathcal{C}_{v_i} \neq \emptyset$ **then**
    $c = \min_{v_i} \mathcal{C}_{v_i}$
    Choose vessel v_i corresponding to c as connection point and $\mathbf{x}_{b_k}$ as bifurcation point.
    $n = n + 1$
  end if
  if $\mathcal{C}_{v_i} = \emptyset$ **then**
    $n = n - 1$
  end if
end while

2.5 Illustration of resulting vascular trees

In all figures below we use the parameter specifications; $\gamma = 3, \delta = 0.7, \nu = 1, f_r = 0.9, N_{con} = 3, \Delta v = 6$ and $N_{fail} = 5$.

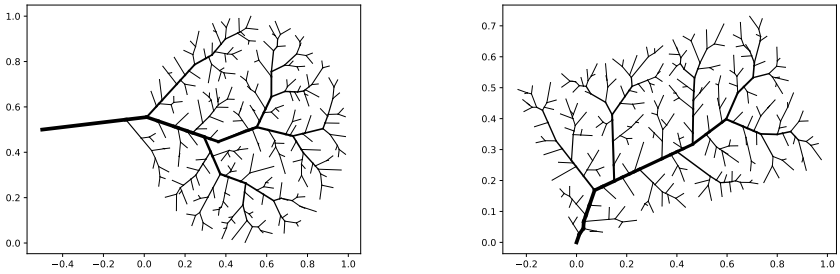
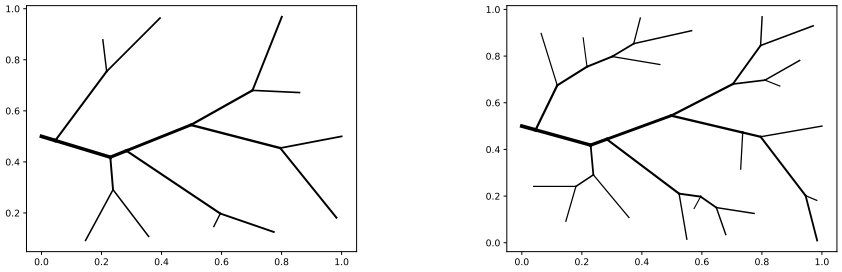
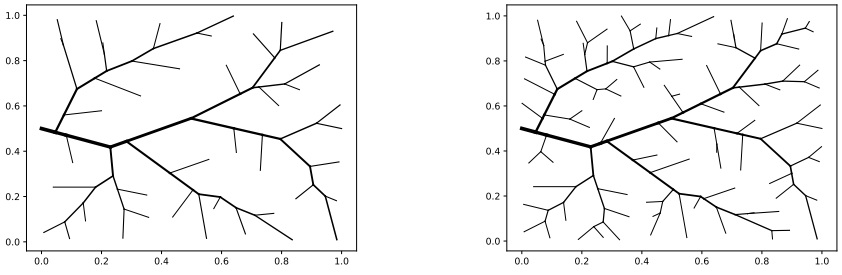


Figure 2.2: Illustration of model capabilities, confined in circle and parallelogram and grown to $N^t_{final} = 200$.



(a) $N^t_{final} = 10$

(b) $N^t_{final} = 20$



(c) $N^t_{final} = 40$

(d) $N^t_{final} = 80$

Figure 2.3: Illustration of tree growth. Root initiated as segment between $(x, y) = (0, 0.5)$ and $(1, 0.5)$ and Ω given as the square with vertices $(0, 0), (1, 0), (1, 1)$ and $(0, 1)$.

CHAPTER 3

Finite element solver for fluid interaction

Having a way to generate vascular trees we now want assign differential equations to the corresponding geometries aiming to describe the interaction of fluid in the vascular trees and the surrounding tissue. This multi-physics problem and the topological and geometric complexity of the vasculature lead to the need for particular attention when deriving governing fluid equations and investigating their numerical stability. We will follow the derivations made in (Cerroni et al., 2019; Masri et al., 2023), which through model reduction of the vasculature obtain well-posed variational formulations of fluid interaction on mixed dimensional domains. Our aim is to provide the necessary details in the first steps of obtaining the mathematical framework for such methods and to provide insight into the corresponding numerical procedure.

First, we provide the necessary tools of functional analysis to establish the existence and uniqueness of a weak solution to a boundary value problem, befor we then embark on deriving a mathematical model for the interaction of a two-dimensional porous medium with the flow through conduits. With model reduction techniques and simplifications, we obtain a well-posed weak formulation of a coupled elliptic boundary value problem describing the interaction of flow through vascular trees and the surrounding tissue. In the framework of the Galerkin finite element method we establish a priori error bounds and by the method of manufactured solution we investigate how different geometries allow for different solutions and in turn different convergence rates. In this way, we establish confidence in the proposed numerical solution of the mathematical problem before we incorporate the vascular trees obtained by the method of CCO. Lastly, we physically motivate the choice of boundary conditions and interpret the results of the numerical solver.

The proposed method provides pressure distributions in vascular trees and the surrounding tissue. Using the method of CCO and this finite element solver we can numerically generate the data needed for the machine learning model.

3.1 Preliminaries

In the mathematical branch of functional analysis one mainly studies functions defined on vector spaces and for partial differential equations a vector space of particular interest is the Hilbert space. To understand why, we express different properties of Hilbert spaces through a series of theorems. In the following, leading up to Theorem 3.1.6 we follow the derivations made in (van Neerven, 2022).

Proposition 3.1.1. *We claim that the orthogonal complement of any subspace X of a Hilbert space H is closed.*

We have

$$X^{\perp} := \{\mathbf{x} \in H | \mathbf{x} \perp \mathbf{y} \quad \forall \mathbf{y} \in X\}$$

and want to show that $X^{\perp}$ is closed.

Proof. Let $\mathbf{x}_n \to \mathbf{x}$ in H with $\mathbf{x}_n \in X^{\perp}$. Then by continuity of the inner product

$$\langle \mathbf{x}, \mathbf{y} \rangle = \lim_{n\to\infty} \langle \mathbf{x}_n, \mathbf{y} \rangle = 0.$$

Whence, we see that $X^{\perp}$ contains all its limit points which means it is a closed subspace of H. ■

Theorem 3.1.2 (Best approximation theorem)**.** *Let C be a nonempty convex closed subspace of H. Then for all $\mathbf{x} \in H$ there exists a unique $\mathbf{c} \in C$ that minimizes the distance from $\mathbf{x}$ to the points of C:*

$$\|\mathbf{x} - \mathbf{c}\| = \min_{\mathbf{y}\in C} \|\mathbf{x} - \mathbf{y}\|$$

Proof. Let $\{\mathbf{y}_n\}_{n\geq 1}$ be a sequence in C such that

$$\lim_{n\to\infty} \|\mathbf{x} - \mathbf{y}_n\| = \inf_{\mathbf{y}\in C} \|\mathbf{x} - \mathbf{y}\| := D$$

By the parallelogram law applied to the vectors $\mathbf{x} - \mathbf{y}_m$ and $\mathbf{x} - \mathbf{y}_n$ we have

$$\|\mathbf{y}_n - \mathbf{y}_m\|^2 + \|2\mathbf{x} - (\mathbf{y}_n - \mathbf{y}_m)\|^2 = 2\|\mathbf{x} - \mathbf{y}_m\|^2 + 2\|\mathbf{x} - \mathbf{y}_n\|^2.$$

As $m, n \to \infty$, the right hand side tends to $2D^2 + 2D^2 = 4D^2$ and from convexity we have $\frac{1}{2}(\mathbf{y}_m - \mathbf{y}_n) \in C$. It follows that

$$\|2\mathbf{x} - (\mathbf{y}_n - \mathbf{y}_m)\|^2 = 4\|\mathbf{x} - \frac{1}{2}(\mathbf{y}_n - \mathbf{y}_m)\|^2 \geq 4D^2.$$

Hence,

$$\begin{aligned}\lim_{m,n\to\infty} \|\mathbf{y}_n - \mathbf{y}_m\|^2 &= 4D^2 - \lim_{n,m\to\infty} \|2\mathbf{x} - (\mathbf{y}_n - \mathbf{y}_m)\|^2 \\ &\leq 4D^2 - 4D^2 = 0.\end{aligned}$$

This shows that the sequence $\{\mathbf{y}_n\}_{n\geq 1}$ is Cauchy in C. Since H is complete we have $\lim_{n\to\infty} \mathbf{y}_n = \mathbf{c}$ for some $\mathbf{c} \in H$, and since C is closed we have $\mathbf{c} \in C$. Now,

$$\|\mathbf{x} - \mathbf{c}\| = \lim_{n\to\infty} \|\mathbf{x} - \mathbf{y}_n\| = D,$$

so $\mathbf{c}$ minimizes the distance to $\mathbf{x}$. ■

Theorem 3.1.3. *If X is a closed linear subspace of H, then we have an orthogonal direct sum decomposition*

$$H = X \oplus X^{\perp}.$$

That is, we have $X \cap X^{\perp} = \{0\}$, $X + X^{\perp} = H$ and $X \perp X^{\perp}$.

Proof. From Proposition 3.1.1 we have that $Y^{\perp}$ is a closed subspace of H. If $\mathbf{y} \in Y \cap Y^{\perp}$ then $\mathbf{y} \perp \mathbf{y}$, so $\langle \mathbf{y}, \mathbf{y} \rangle = 0$ and $\mathbf{y} = 0$. It remains to show that $Y + Y^{\perp} = H$. Let $\mathbf{x} \in H$ be arbitrary and fixed, we must show that $\mathbf{x} \in Y + Y^{\perp}$. Let $f_y : H \to X$ denote the mapping arising from the previous theorem. Then $f_y(\mathbf{x})$ is the unique element of Y that minimizes the distance to $\mathbf{x}$ i.e.

$$\|\mathbf{x} - f_y(\mathbf{x})\| = \min_{\mathbf{y} \in Y} \|\mathbf{x} - \mathbf{y}\|.$$

Set $\mathbf{y}_0 := f_y(\mathbf{x})$ and $\mathbf{y}_1 := \mathbf{x} - \mathbf{y}_0$. Then $\mathbf{y}_0 \in Y$, and for all $\mathbf{y} \in Y$ we have

$$\begin{aligned} \|\mathbf{y}_1\| = \|\mathbf{x} - \mathbf{y}_0\| &\leq \|\mathbf{x} - (\mathbf{y}_0 - \mathbf{y})\| \\ &= \|\mathbf{y} + (\mathbf{x} - \mathbf{y}_0)\| = \|\mathbf{y} + \mathbf{y}_1\|. \end{aligned}$$

We now claim that this implies $\mathbf{y}_1 \in Y^{\perp}$, to see this, fix a nonzero $\mathbf{y} \in Y$. For any $c \in \mathbb{F}^1$ we have

$$\|\mathbf{y}_1\|^2 \leq \|c\mathbf{y} + \mathbf{y}_1\|^2 = |c|^2 \|\mathbf{y}\|^2 + 2\mathrm{Re}\langle c\mathbf{y}, \mathbf{y} \rangle + \|\mathbf{y}_1\|^2$$

Taking $c = -\frac{\overline{\langle \mathbf{y}, \mathbf{y}_1 \rangle}}{\|\mathbf{y}\|^2}$ gives

$$0 \leq \frac{\langle \mathbf{y}, \mathbf{y}_1 \rangle^2}{\|\mathbf{y}\|^2} - 2\frac{\langle \mathbf{y}, \mathbf{y}_1 \rangle^2}{\|\mathbf{y}\|^2} = -\frac{\langle \mathbf{y}, \mathbf{y}_1 \rangle^2}{\|\mathbf{y}\|^2}$$

where the left-hand side of the above is only possible if $\langle \mathbf{y}, \mathbf{y}_1 \rangle = 0$. Since $\mathbf{y} \neq 0 \in Y$ was arbitrary this shows that $\mathbf{y}_1 \in Y^{\perp}$ and proves the claim. It follows that $\mathbf{x} = \mathbf{y}_0 + \mathbf{y}_1$ belongs to $Y + Y^{\perp}$. ■

Theorem 3.1.4 (Riesz representation theorem)**.** *If T: $H \to \mathbb{F}$ is a bounded linear functional, there exists a unique element $\mathbf{y} \in H$ such that*

$$T(\mathbf{x}) = \langle \mathbf{y}, \mathbf{x} \rangle \qquad \forall \ \mathbf{x} \in H. \tag{3.1}$$

Proof. If $T(\mathbf{x}) = 0 \ \forall \mathbf{x} \in H$, we take $\mathbf{y} = 0$. So assume $T(\mathbf{x}) \neq 0$, we know from Theorem 3.1.3 that closed subspaces of a Hilbert space H is orthogonally complemented. Whence $(N(T))^{\perp} \neq \{0\}$ [2] so we can choose a norm one vector $\mathbf{y}_0 \in (N(T))^{\perp}$ and fix an arbitrary $\mathbf{x} \in H$ with $c := \frac{T(x)}{T(y_0)}$. We then have

$$T(\mathbf{x} - c\mathbf{y}_0) = T(\mathbf{x}) - cT(\mathbf{y}_0) = 0$$

where we have used the linearity of T and our definition of c. This means that $\mathbf{x} - c\mathbf{y}_0 \in N(T)$, so $\mathbf{x} - c\mathbf{y}_0 \perp \mathbf{y}_0$ and

$$T(\mathbf{x}) = cT(\mathbf{y}_0) = T(\mathbf{y}_0)\langle c\mathbf{y}_0, \mathbf{y}_0 \rangle = T(\mathbf{y}_0)\langle \mathbf{x}, \mathbf{y}_0 \rangle = \langle \mathbf{x}, \overline{T(\mathbf{y}_0)}\mathbf{y}_0 \rangle$$

[1] $\mathbb{F}$ is the field consisting of all real $\mathbb{R}$ and complex $\mathbb{C}$ numbers.
[2] $N(T)$ denotes the null space of T, i.e. $N(T) = \{\mathbf{x} \in H : T(\mathbf{x}) = 0\}$.

Setting $\mathbf{y} = \overline{T(\mathbf{y}_0)}\mathbf{y}_0$ we obtain (3.1). To prove uniqueness, suppose that $T = T(\mathbf{y}) = T(\mathbf{y}')$ for $\mathbf{y}, \mathbf{y}' \in H$. Then

$$\|\mathbf{y}-\mathbf{y}'\|^2 = \langle \mathbf{y}-\mathbf{y}', \mathbf{y}-\mathbf{y}' \rangle = \langle \mathbf{y}-\mathbf{y}', \mathbf{y} \rangle - \langle \mathbf{y}-\mathbf{y}', \mathbf{y}' \rangle = T(\mathbf{y}-\mathbf{y}') - T(\mathbf{y}-\mathbf{y}') = 0,$$

where we have used the properties of the inner product and (3.1). Since the norm is positive definite we see that the above derivation implies $\mathbf{y} = \mathbf{y}'$. ■

Proposition 3.1.5. *Assume $A : H \to H$ to be a bounded linear operator that is bounded from below i.e.*

$$\|A\mathbf{x}\| \geq C\|\mathbf{x}\|$$

for $C > 0$ and all $\mathbf{x} \in H$. Then A is injective (1-1) and has closed range.

Proof. Assume A is not 1-1, then $A\mathbf{x} = A\mathbf{y}$ for $\mathbf{x} \neq \mathbf{y}$, but since A is bounded from below we have

$$0 = \|A\mathbf{x} - A\mathbf{y}\| = \|A(\mathbf{x} - \mathbf{y})\| \geq C\|\mathbf{x} - \mathbf{y}\| > 0$$

which contradicts $\mathbf{x} \neq \mathbf{y}$. To show that $R(A)$ is closed let $\{\mathbf{y}_n\}_{n\geq 1}$ be a sequence in $R(A)$ that converges to a point $\mathbf{y} \in H$. For each $n \in \mathbb{N}$, there exists $\mathbf{x}_n \in H$ such that $\mathbf{y}_n = A(\mathbf{x}_n)$. For any $m, n \in \mathbb{N}$ we have

$$C\|\mathbf{x}_m - \mathbf{x}_n\| \leq \|A(\mathbf{x}_m - \mathbf{x}_n)\| = \|\mathbf{y}_m - \mathbf{y}_n\|.$$

Since $\{\mathbf{y}_n\}_{n\geq 1}$ is a Cauchy sequence, we see from the above derivation that $\{\mathbf{x}_n\}_{n\geq 1}$ is also a Cauchy sequence. Since H is a Banach space it is complete and there exists a $\mathbf{x} \in H$ s.t. $\mathbf{x}_n \to \mathbf{x}$. Since A is bounded it is continuous and $\mathbf{y} = A\mathbf{x} \in R(A)$. ■

Now, we are ready to introduce the main theorem for the well-posedness of elliptic problems. The theorem is stated as in (Evans, 1998). We now use lower case letters to denote functions, taking either a finite or infinite number of values.

Theorem 3.1.6 (Lax Milgram theorem)**.** *Assume that $B : H \times H \to \mathbb{R}$ is a bilinear mapping, for which there exists constants $\alpha, \beta > 0$ such that*

i) $|B[u, v]| \leq \alpha\|u\|\|v\|$
ii) $\beta\|u\|^2 \leq B[u, u]$

where $u, v \in H$. Finally let $F : H \to \mathbb{R}$ be a bounded linear functional on H. Then there exists a unique $u \in H$ such that

$$B[u, v] = F(v)$$

for all $v \in H$.

Proof. For each fixed $u \in H$, the mapping $v \mapsto B[u, v]$ is a bounded linear functional on H, whence by Theorem 3.1.4 exists a unique element $w \in H$ satisfying

$$B[u, v] = \langle w, v \rangle.$$

Let us write $Au = w$ whenever the above holds, so that

$$B[u, v] = \langle Au, v \rangle$$

for $u, v \in H$. We claim that $A : H \to H$ is a bounded linear operator, 1-1 and that the range of A is closed in H.
To prove that A is linear we let $\lambda_1, \lambda_2 \in \mathbb{R}$ and $u_1, u_2 \in H$. We then have

$$\begin{aligned}\langle A(\lambda_1 u_1 + \lambda_2 u_2, v\rangle &= B[\lambda_1 u_1, \lambda_2 u_2, v] \\ &= \lambda_1 B[u_1, v] + \lambda_2 B[u_2, v] \\ &= \lambda_1 \langle Au_1, v\rangle + \lambda_2 \langle Au_2, v\rangle \\ &= \langle \lambda_1 Au_1 + \lambda_2 Au_2, v\rangle.\end{aligned}$$

Since this holds for all $v \in H$, A is a linear operator. Furthermore,

$$\|Au\|^2 = \langle Au, Au\rangle = B[u, Au] \leq \alpha \|u\| \|Au\|$$

hence,

$$\|Au\| \leq \alpha \|u\|.$$

So A is bounded. Next, we compute

$$\beta \|u\|^2 \leq B[u, u] = \langle Au, u\rangle \leq \|Au\| \|u\|.$$

Where in the last inequality we have used the Cauchy-Schwarz inequality (see e.g. Evans, 1998, page. 708). This shows that A is bounded from below

$$\beta \|u\| \leq \|Au\|$$

and by Proposition 3.1.5 we have that A is 1-1 and $R(A)$ is closed. We demonstrate now that $R(A) = H$. For if not, then, since $R(A)$ is closed, there would by Theorem 3.1.3 exists a nonzero element $w \in H$ with $w \in R(A)^{\perp}$. This fact in turn implies

$$\beta \|w\|^2 \leq B[w, w] = \langle Aw, w\rangle = 0$$

So, w must be zero, and we have obtained a contradiction.
Next, we observe once more from Theorem 3.1.4 that

$$F(v) = \langle w, v\rangle \qquad \forall v \in H$$

for some element $w \in H$. We can then use our previous derivations to find $u \in H$ satisfying $w = Au$, then

$$B[u, v] = \langle Au, v\rangle = \langle w, v\rangle = F(v).$$

Finally, we show that there is at most one element $u \in H$ satisfying the above. For if both

$$B[u_1, v] = F(v) \qquad \text{and} \qquad B[u_2, v] = F(v)$$

then $B[u_1 - u_2, v] = 0 \ \forall v \in H$. We set $v = u_1 - u_2$ to find

$$\beta \|u_1 - u_2\|^2 \leq B[u_1 - u_2, u_1 - u_2] = 0 \to u_1 = u_2$$

■

Sobolev spaces and operators

We define the following spaces and operators which will be used in this chapter. Let $\Omega \in \mathbb{R}^n$, then the Sobolev space $W^{m,p}(\Omega)$ consist of all functions u on Ω such that for each multi-index $|\alpha| \leq m$ the weak or distrubutional derivative

$$D^\alpha = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \cdots \partial x_n^{\alpha_n}}$$

exists and $\|D^\alpha u\|_{L^p(\Omega)} < \infty$. The L^p-norm is defined as

$$\|u\|_{L^p(\Omega)} := \left(\int_\Omega |u|^p dx \right)^{\frac{1}{p}}. \tag{3.2}$$

When $p = 2$ the Sobolev space $W^{m,2}$ becomes a Hilbert space and we denote it H^m,

$$H^m := \{u \in L^2 : \|D^m u\|_{L^2} < \infty \; \forall |\alpha| < \infty\}.$$

This Hilbert space is equipped with the norm

$$\|u\|_{H^m(\Omega)} := \left(\sum_{|\alpha| \leq m} \|D^m u\|_{L^2(\Omega)} \right)^{\frac{1}{2}}$$

and if u is differentiable in the classical sence then the weak and strong derivative coincide, i.e. we have $\nabla^m = D^m$.
Since functions $u \in H^1(\Omega)$ only need to be defined *almost everywhere* in Ω we need to specify what we mean by $u|_{\partial\Omega}$. Let Ω be a bounded domain with Lipschitz boundary $\partial\Omega$, we then define the trace operator as the operator

$$T : H^1(\Omega) \to L^2(\partial\Omega) \tag{3.3}$$

which maps functions in $H^1(\Omega)$ to $L^2(\partial\Omega)$. We have the properties

$$Tu = u|_{\partial\Omega} \qquad \text{if } u \in H^1 \cap C(\bar{\Omega}) \tag{3.4}$$

$$\|Tu\|_{L^2(\partial\Omega)} \leq C_T \|u\|_{H^1(\Omega)} \qquad \text{if } u \in H^1(\Omega) \tag{3.5}$$

where $\bar{\Omega}$ denotes the closure of Ω, i.e. if u can be continuously extended to the boundary $\partial\Omega$ then the trace of u is equal to u evaluated at the boundary. For the existence of such an operator we refer to (Evans, 1998, p. 272).

We also define the fractional Sobolev space $H^{\frac{1}{2}}(\partial\Omega)$ as the range of the trace operator

$$H^{\frac{1}{2}}(\partial\Omega) := \{g \in L^2(\partial\Omega) : \exists Eg \in H^1(\Omega) : T(Eg) = g\}. \tag{3.6}$$

With this definiton we have the exstension operator $E : H^{\frac{1}{2}}(\partial\Omega) \to H^1$ as the operator taking the "boundary data" $g \in H^{\frac{1}{2}}(\partial\Omega)$ and exstending it to Ω with regularity $H^1(\Omega)$. For the proof of the existence of such an operator, we refer to (Grisvard, 2011, theorem 1.5.1.3, 1.5.2.1). Lastly, we denote with the subscript $H^1_\alpha(\Omega)$ the spaces

$$H^1_\alpha(\Omega) = \{H^1(\Omega) : u = \alpha \text{ on } \partial\Omega\} \tag{3.7}$$

where the function u takes spesific values on the boundary. With the definitions above we are ready to start building the mathematical model.

3.2 The 2D-1D boundary value problem

Our aim is to build a model of the interaction of flow in the vasculature and the surrounding tissue. To derive the proposed model we first follow (Cerroni et al., 2019) describing the interaction of wells with reservoirs by means of dimensionality reduction, reducing a 3D-3D problem to a 3D-1D problem. We will follow the proposed model reduction technique but in a 2D-2D to 2D-1D setting. The 2D domain then represents the tissue/reservoir and the 2D domain reduced to 1D, the vasculature/well. We first consider a single vessel, before we extend the problem to hold for vascular trees.

Following the notation used in (Cerroni et al., 2019) we let Ω_p denote the, from now on called, tissue, Ω_w denote the, from now on called, vessel and Γ the interface between Ω_p and Ω_w. We require Ω_w to be a C^2-regular curve. Let $\mathbf{v}_p, u_p$ denote the flow velocity and pressure in the tissue and $\mathbf{v}_w, u_w$ the velocity and pressure *injected* or *extracted* through the vessel. We model the tissue as a porous medium, and thus $\mathbf{v}_p, u_p$ follows Darcy's law (Whitaker, 1986)

$$\mathbf{v} = -\mathbf{k}\nabla u \tag{3.8}$$

where $\mathbf{k}$ is the permeability tensor. Motivated by our derivation in chapter 2 and specifically Poiseuville's law (2.3) we also model $\mathbf{v}_w$ as a flow velocity purely driven by the pressure gradient, and thus also follows (3.8). Next, we presume that the compressibility of $\mathbf{v}_*$, $* = p, w$ is small enough to be neglected, leading to the incompressibility condition

$$\nabla \cdot \mathbf{v} = 0. \tag{3.9}$$

Lastly, we presume that the interface Γ is permeable, and model this by a *normal* flux $k_\Gamma(u_p - u_w)$ for flow "leaking" from the tissue to the vessel and equal but opposite from the vessel to the tissue. $k_\Gamma \geq 0$ denotes the permeability of the interface Γ. Summarizing the above we have

$$\begin{cases} \nabla \cdot \mathbf{v}_p = 0, \quad \mathbf{v}_p + \mathbf{k}_p \nabla u_p = 0 & \text{in } \Omega_p \\ \nabla \cdot \mathbf{v}_w = 0 \quad \mathbf{v}_w + \mathbf{k}_w \nabla u_w = 0 & \text{in } \Omega_w \\ \mathbf{v}_p \cdot \mathbf{n}_p = k_\Gamma(u_p - u_w) & \text{on } \Gamma \\ \mathbf{v}_w \cdot \mathbf{n}_w = k_\Gamma(u_w - u_p) & \text{on } \Gamma. \end{cases} \tag{3.10}$$

The above equations can be written as a function of the pressures solely by combining (3.8) and (3.9). Furthermore, we consider the pressures dimensionless quantities and presume constant isotropic permeability tensors $k_* = k_* \mathbf{I}$. with $k_* > 0$. We then have

$$\begin{cases} -k_p \Delta u_p = 0 & \text{in } \Omega_p \\ -k_w \Delta u_w = 0 & \text{in } \Omega_w \\ -k_p \nabla u_p \cdot \mathbf{n}_p = k_\Gamma(u_p - u_w) & \text{on } \Gamma \\ -k_w \nabla u_w \cdot \mathbf{n}_w = k_\Gamma(u_w - u_p) & \text{on } \Gamma. \end{cases} \tag{3.11}$$

To make the above boundary value problem complete we need to add boundary conditions and motivated by the fact that we later will use manufactured solutions enforced through Dirichlet boundary conditions we choose

$$u_p = g_p \quad \text{on } \Sigma \tag{3.12}$$

$$u_w = g_w \quad \text{on } \partial\Omega_w \tag{3.13}$$

where $\Sigma = \partial\Omega_p/\Gamma$ is a subset of the boundary $\partial\Omega$, and $\partial\Omega_w$ is the starting point and endpoint of the vascular segment. The specific choice of g_w will be specified later, but since Ω_w will be reduced to one dimension we let g_w be constant on the two respective parts of $\partial\Omega_w$. We also require $g_p \in H^{\frac{1}{2}}(\Sigma)$ and $g_w \in H^{\frac{1}{2}}(\partial\Omega_w)$.

Model reduction of the vessel

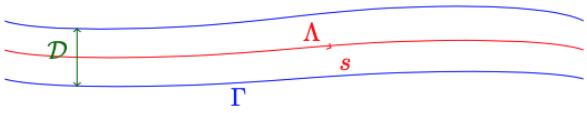


Figure 3.1: Example of a portion of the vessel used for model reduction, surrounded by the tissue Ω_p.

Consider a portion $\mathcal{P}$ of the vessel Ω_w as depicted in Figure 3.1, we parametrize the centerline Λ by

$$\Lambda = \{\boldsymbol{\lambda}(s), s \in (s_1, s_2)\}$$

and cross-section by

$$\mathcal{D}(s) = [-r(s)\mathbf{n}(s), r(s)\mathbf{n}(s)] : [0, R) \to \mathbb{R}^2$$

where $\mathbf{n}$ is the unit normal perpendicular to Λ and r is the distance perpendicular from the interface. We then have the parametrization of the portion $\mathcal{P}$ as

$$\mathcal{P} = \{\lambda(s) - r(s)\mathbf{n}(s) + r(s)\mathbf{n}(s), s \in (s_1, s_2), r \in [0, \rho)\} \tag{3.14}$$

with the interface

$$\Gamma = \{\lambda(s) - R(s)\mathbf{n}(s) + R(s)\mathbf{n}(s), s \in (s_1, s_2), R \in [0, \rho)\} \tag{3.15}$$

such that $2R$ is the width of the vascular segment at location s along the interface. We now want to apply a topological model reduction such that the depiction above can be expressed as a 2D-1D problem. We make the following assumptions

1. The radius of the vascular segment is small compared to the size of the tissue, in particular, $R/L \ll 1.$, where L loosly represents the length *and* width of the surrounding tissue.

2. With the first assumption in mind, we presume further that u_w has a uniform profile on each cross-section $\mathcal{D}$, i.e. u_w can be written as a function of s only and we let $\hat{u}(s) = u_w(r, s)$ denote the one-dimensional representation of u_w.

Let w be an arbitrary function and let us introduce the following notation

$$\bar{\bar{w}}(s) = \frac{1}{2R}\int_{\mathcal{D}(s)} w d\sigma \tag{3.16}$$

is the mean of w over $\mathcal{D}$ at a specific location s on the centerline such that

$$\int_\Lambda \int_{\mathcal{D}} w d\sigma ds = \int_\Lambda 2R\bar{\bar{w}}(s) ds. \tag{3.17}$$

Similarly, we let

$$\bar{w}(s) = \frac{1}{2} w|_{\partial\mathcal{D}} \tag{3.18}$$

be the mean of w on $\partial\mathcal{D}$ such that

$$\int_\Lambda w|_{\partial\mathcal{D}} ds = \int_\Lambda 2\bar{w}(s) ds. \tag{3.19}$$

Integrating the second term in (3.11) over $\mathcal{P}$ we have

$$\begin{aligned} -k_w \int_{\mathcal{P}} \Delta u_w dx &= -\int_{\partial\mathcal{P}} k_w \nabla u_w \cdot \mathbf{n}_w \\ &= \int_{\mathcal{D}(s_1)} \frac{\partial}{\partial s} k_w u_w d\sigma - \int_{\mathcal{D}(s_2)} \frac{\partial}{\partial s} k_w u_w d\sigma - \int_\Gamma k_w \nabla u_w \cdot \mathbf{n}_w d\sigma \end{aligned} \tag{3.20}$$

where the first equality is from the divergence theorem. Applying the fundamental theorem of calculus we can rewrite the first two terms in the second equality as

$$\begin{aligned} \int_{\mathcal{D}(s_1)} \frac{\partial}{\partial s} k_w u_w d\sigma - \int_{\mathcal{D}(s_2)} \frac{\partial}{\partial s} k_w u_w d\sigma &= -\int_{s_1}^{s_2} \frac{d}{ds}\left(\int_{\mathcal{D}} \frac{\partial}{\partial s} k_w u_w d\sigma\right) ds \\ &= -\int_{s_1}^{s_2} \frac{d}{ds}\left(2k_w R \frac{\partial}{\partial s}\bar{\bar{u}}_w\right) ds \end{aligned} \tag{3.21}$$

where in the last inequality we have used (3.17) and our assumption of constant permeability k_w. Next, we integrate the interface condition given last in (3.11) to obtain

$$\begin{aligned} -\int_\Gamma k_w \nabla u_w \cdot \mathbf{n}_w &= \int_\Gamma k_\Gamma (u_w - u_p) \\ &= \int_{s_1}^{s_2} [k_\Gamma (u_w - u_p)]_{\partial\mathcal{D}} ds = \int_{s_1}^{s_2} 2k_\Gamma (\bar{u}_w - \bar{u}_p) ds. \end{aligned} \tag{3.22}$$

Inserting (3.22) and (3.21) in (3.20) and recalling the left hand side in (3.11) we obtain

$$\int_{s_1}^{s_2} \frac{d}{ds}\left(2k_w R \frac{\partial}{\partial s}\bar{\bar{u}}_w\right) + 2k_\Gamma(\bar{u}_w - \bar{u}_p) ds = 0. \tag{3.23}$$

Since the choice of s_1 and s_2 was arbitrary we conclude

$$\frac{d}{ds}\left(2k_w R \frac{\partial}{\partial s}\bar{\bar{u}}_w\right) + 2k_\Gamma(\bar{u}_w - \bar{u}_p) = 0. \tag{3.24}$$

Recalling our assumption that u_w has uniform profile, we can insert $\hat{u}$ for the averages above and obtain

$$\frac{d}{ds}\left(2k_w R\frac{\partial}{\partial s}\hat{u}\right) + 2k_\Gamma(\hat{u} - \bar{u}_p) = 0 \text{ on } \Lambda, \tag{3.25}$$

since $\hat{u}$ is a single variable function of s the partial derivative is the total derivative and we let $\Delta\hat{u} = \frac{d^2}{ds^2}\hat{u}$ to obtain

$$2k_w R\Delta\hat{u} + 2k_\Gamma(\hat{u} - \bar{u}_p) = 0 \text{ on } \Lambda. \tag{3.26}$$

Lastly, including the boundary conditions in (3.13) we get the boundary value problem on the center line Λ

$$2k_w R\Delta\hat{u} + 2k_\Gamma(\hat{u} - \bar{u}_p) = 0 \quad \text{on} \quad \Lambda \tag{3.27}$$

$$\hat{u} = \hat{g} \quad \text{on } \partial\Lambda \tag{3.28}$$

where we have used the same notation on the boundary term i.e. $\hat{g} = g_w$ to emphasize the relation to $\hat{u}$.

Model reduction of the interface

Let w on Γ be expressed as its average plus some fluctuation

$$w = \bar{w} + \tilde{w}. \tag{3.29}$$

Then we make the following assumptions

3. We assume that the product of fluctuations is small over each interface $\partial\mathcal{D}$, if w_1, w_2 are functions in (Ω_p, Ω_w) then

$$(\tilde{w}_1\tilde{w}_2)|_{\partial\mathcal{D}} = 0.$$

Furthermore, the mean of the fluctuation is zero, $\bar{\tilde{w}} = \tilde{w}|_{\partial\mathcal{D}} = 0$.

4. We identify Ω_p with the entire $\Omega = \Omega_p \cup \Omega_w$ and hence skip the subscript p.

Multiplying the first term in (3.11) with test functions v and integrating over the domain we get

$$-\int_{\Omega_p}(k_p\Delta u_p)v = \int_\Omega k\nabla u\cdot\nabla v dx - \int_{\partial\Omega}(k\nabla u\cdot\mathbf{n})v dS \tag{3.30}$$

where we have used integration by parts and Green's identity. Next, we split the boundary integral into its respective parts

$$\int_{\partial\Omega}(k\nabla u\cdot\mathbf{n})v ds = \int_\Sigma(k\nabla u\cdot\mathbf{n})v dS + \int_\Gamma(k\nabla u\cdot\mathbf{n})v dS \tag{3.31}$$

and the integral over the Dirichlet boundary vanishes by choosing the appropriate test space $H^1_{\Sigma_0} = \{v \in H^1(\Omega_p) : v = 0 \text{ on } \partial\Sigma\}$. From the interface condition in (3.11) we have

$$-\int_\Gamma(k_p\nabla u_p\cdot\mathbf{n}_p)v ds = \int_\Gamma k_\Gamma(u_p - u_w)v ds = \int_\Gamma k_\Gamma u_p v ds - \int_\Gamma k_\Gamma u_w v ds. \tag{3.32}$$

where the last terms can be written as the evaluation of the functions on $\partial\mathcal{D}$ and integrated over the centerline Λ,

$$\int_\Gamma k_\Gamma u_* v ds = \int_\Lambda k_\Gamma(s)(u_* v)|_{\partial\mathcal{D}} ds \tag{3.33}$$

for $* = w, p$. Inserting (3.29) we have

$$\int_\Lambda k_\Gamma(s)(u_* v)|_{\partial\mathcal{D}} ds = \int_\Lambda k_\Gamma(s)\big((\bar{u}_* + \tilde{u}_*)(\bar{v} + \tilde{v})\big)|_{\partial\mathcal{D}} ds \tag{3.34}$$

$$= \int_\Lambda k_\Gamma(s)\big(\bar{u}_*\bar{v} + \bar{u}_*\tilde{v} + \tilde{u}_*\bar{v} + \tilde{u}_*\tilde{v}\big)|_{\partial\mathcal{D}} ds, \tag{3.35}$$

by assumption 3 and noting that the means is only a function of s we obtain

$$\int_\Lambda k_\Gamma(s)(u_* v)|_{\partial\mathcal{D}} ds = \int_\Lambda 2k_\Gamma \bar{u}_*\bar{v} ds. \tag{3.36}$$

Putting together (3.30), (3.36), the right hand side of (3.11) and using assumption 4. we obtain

$$\int_\Omega k\nabla u \cdot \nabla v dx + \int_\Lambda 2k_\Gamma(\bar{u} - \hat{u})\bar{v} ds = 0 \;\; \forall v \in H^1_{\Sigma_0}(\Omega). \tag{3.37}$$

Extension to vascular tree

The previous derivations model the interaction of fluid in one single vessel and the surrounding tissue. Using conservation of fluxes and continuity through bifurcations we can extend the model to vascular trees. Before we begin the derivation we simplify by assuming that

6. The radii throughout the vascular tree is constant, $R_i = R \; \forall i \in N$, and independent of s.

7. We have the relation $k_p = 2Rk_w$ such that the only free parameter is the "coupling factor" $\gamma = \frac{2k_\Gamma}{k_p} = \frac{2k_\Gamma}{2Rk_w}$, $\gamma \geq 0$.

Let Λ be constructed as by the method of CCO, then Λ can be expressed as the union of the vascular segments $\Lambda = \cup_{i=1}^N \Lambda_i$, where $\Lambda_i = \{\boldsymbol{\lambda}_i(s_i), s_i \in (0, S_i)\}$. Since each vascular segment was presumed to be C^2 curves we know that (3.28) and (3.37) holds on each segment, specifically that $\Delta\hat{u}$ is well-defined on each Λ_i. We then let

$$\mathcal{B} = \{\mathbf{b}_j \in \mathbb{R}^2 : j = 1, ..., M\}$$

be the set of all bifurcation points in Λ and for a specific bifurcation point $\mathbf{b}_j$ we have

$$k = \{i \in \{1, ..., N\} : \boldsymbol{\lambda}_i(0) = \mathbf{b}_j\}$$
$$O_j = \{i \in \{1, ..., N\} : \boldsymbol{\lambda}_i(S_i) = \mathbf{b}_j\}$$

where k represents the index of the proximal vessels to the bifurcation and O_j the indices of the two vessels distal to the bifurcation. Furthermore, we let

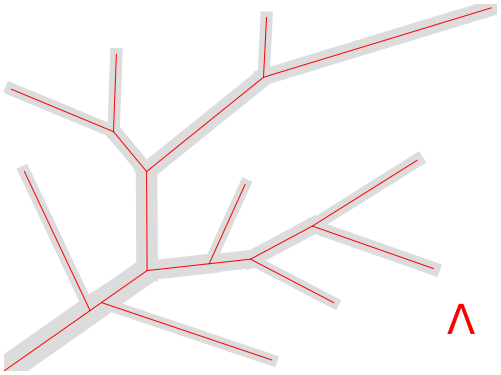


Figure 3.2: Example of model reduction on vascular tree.

$\hat{u}_i = 1, ..., N$ denote the solution $\hat{u}$ on each segment Λ_i. Following (Masri et al., 2023) and (Laurino & Zunino, 2019) we require

$$(\frac{d}{ds_k}\hat{u}_k)(0) = \sum_{l \in O_j} (\frac{d}{ds_l}\hat{u}_l)(S_l) \tag{3.38}$$

$$\hat{u}_k(0) = \hat{u}_l(S_l) \qquad \forall l \in O_j. \tag{3.39}$$

(3.38) corresponds to a balance of fluxes in the bifurcations points, and (3.39) corresponds to continuity. For the root ($i = 0$) and all terminal vessels ($t = 1, ..., N_{term}$) we express the Dirichlet boundary conditions by letting

$$\hat{u}_0(0) = d_0, \qquad \hat{u}_t(S_t) = d_t. \tag{3.40}$$

We define

$$H^1(\Lambda) := \bigoplus_{\Lambda_i} H^1(\Lambda_i) \cap C^0(\Lambda) \tag{3.41}$$

as the Hilbert space consisting of all functions that are locally in H^1 and continuous over each bifurcation point. Furthermore, we define

$$H^1_D := \{v \in H^1(\Lambda) : v_0(0) = d_0, v_t(S_t) = d_t, t = 1, .., N_{term}\}$$
$$H^1_0 := \{v \in H^1(\Lambda) : v_0(0) = 0, v_t(S_t) = 0, t = 1, .., N_{term}\}.$$

Multiplying (3.28) with test functions $v \in H^1_0(\Lambda)$, using the simplifications and integrating we obtain

$$\sum_{i=1}^{N} \int_{\Lambda_i} \nabla\hat{u}_i \cdot \nabla\hat{v}_i ds - [\nabla\hat{u}_i\hat{v}_i]_{\partial\lambda_i} + \gamma \int_{\Lambda_i} (\hat{u}_i - \bar{u})\hat{v}_i ds = 0,$$

where ∇, in this case, is the derivative w.r.t. s_i. Using the notation introduced earlier we can rewrite the boundary term as

$$\sum_{i=1}^{N} [\nabla\hat{u}_i\hat{v}_i]_{\partial\Lambda_i} = [\nabla\hat{u}_0\hat{v}_0]_{\partial\Lambda_0} + \sum_{j=1}^{M} \left[[\nabla\hat{u}_k\hat{v}_k]_{\partial\Lambda_k} - \sum_{i \in O_j} [\nabla\hat{u}_i\hat{v}_i]_{\partial\Lambda_i} \right]$$

$$+ \sum_{t=1}^{N_{term}} [\nabla \hat{u}_t \hat{v}_t]_{\partial\Lambda_t}.$$

By the definition of the test space the first and last terms above are zero, and since the test functions are continuous over each bifurcation we have for each j and each $i \in O_j$, $\hat{v}_k = \hat{v}_i$, hence we can factorize out the test functions to obtain

$$\sum_{i=1}^{N} [\nabla \hat{u}_i \hat{v}_i]_{\partial\Lambda_i} = \sum_{j=1}^{M} \left[[\nabla \hat{u}_k]_{\partial\Lambda_k} - \sum_{i \in O_j} [\nabla \hat{u}_i]_{\partial\Lambda_i} \right] \hat{v}_j$$

which is zero by the balance of fluxes (3.38). Letting $\hat{u} \in H^1_D(\Lambda)$ denote the collection of functions satisfying (3.42) and taking the Dirichlet values (3.40) at the root proximal point and terminal distal points we have the variational formulation; find $\hat{u} \in H^1_D(\Lambda)$ such that

$$\int_\Lambda \nabla \hat{u} \cdot \nabla \hat{v} ds + \gamma \int_\Lambda (\hat{u} - \bar{u}) \hat{v} ds = 0 \qquad \forall \hat{v} \in H^1_0(\Lambda). \tag{3.42}$$

Since R is constant throughout the vascular tree, the average $\bar{u}$ is by definition constant over each cross-section $\mathcal{D}$ and since the averages only appear in the last terms of (3.28) and (3.37) we approximate $\bar{u}$ by the centerline trace of u, i.e. we let $\bar{u} = u|_\Lambda$. Since (3.37) should hold for all vascular segments Λ_i we can combine (3.37) and (3.28) to obtain the boundary value problem of finding $(u, \hat{u}) \in H^1_{\Sigma_D}(\Omega) \times H^1_D(\Lambda)$ such that

$$\int_\Omega \nabla u \cdot \nabla v dx + \int_\Lambda \nabla \hat{u} \cdot \nabla \hat{v} ds + \gamma \int_\Lambda (u - \hat{u})(v - \hat{v}) ds = 0 \ \ \forall (v, \hat{v}) \in H^1_{\Sigma_0}(\Omega) \times H^1_0(\Lambda). \tag{3.43}$$

3.3 Well-posedness

We aim to prove well-posedness of the boundary value problem (3.43) by showing that it satisfies the requirements of Lax-Milgram, Theorem 3.1.6. Since we seek a solution pair $(u, \hat{u})$ of the boundary value problem (3.43) we first note the following theorem.

Theorem 3.3.1. *Let $H^1(\Omega)$ and $H^1(\Lambda)$ be known Hilbert spaces with inner products*

$$\langle u, v \rangle_{H^1(\Omega)} = \int_\Omega \nabla u \cdot \nabla v dx + \int_\Omega uv dx$$
$$\langle \hat{u}, \hat{v} \rangle_{H^1(\Lambda)} = \int_\Lambda \nabla \hat{u} \cdot \nabla \hat{v} ds + \int_\Lambda \hat{u}\hat{v} ds.$$

Then let $u, v \in H^1(\Omega), \hat{u}, \hat{v} \in H^1(\Lambda)$, and denote the Cartesian product of $H^1(\Omega)$ and $H^1(\Lambda)$ by

$$\mathbb{V} = H^1(\Omega) \times H^1(\Lambda).$$

We aim to prove that this is also a Hilbert space and that an appropriate inner product is defined by

$$\langle (u, \hat{u}), (v, \hat{v}) \rangle_{\mathbb{V}} = \langle u, v \rangle_{H^1(\Omega)} + \langle \hat{u}, \hat{v} \rangle_{H^1(\Lambda)},$$

which induces the norm

$$\|(v,\hat{v})\|_{\mathbb{V}}^2 = \|v\|_{H^1(\Omega)}^2 + \|\hat{v}\|_{H^1(\Lambda)}^2. \tag{3.44}$$

Proof. We first check that the proposed inner product is indeed an inner product. we have

$$\begin{aligned}\langle (u,\hat{u}),(v,\hat{v})\rangle &= \langle u,v\rangle + \langle \hat{u},\hat{v}\rangle \\ &= \langle v,u\rangle + \langle \hat{v},\hat{u}\rangle = \langle (v,\hat{v}),(u,\hat{u})\rangle\end{aligned}$$

which shows that the proposed inner product is symmetric. To show linearity we let $a,b \in \mathbb{R}$ and have

$$\begin{aligned}\langle a(u,\hat{u}) + b(v,\hat{v}),(w,\hat{w})\rangle &= \langle (au+bv, a\hat{u}+b\hat{v}),(w,\hat{w})\rangle \\ &= \langle au+bv,w\rangle + \langle a\hat{u}+b\hat{v},\hat{w}\rangle \\ &= a\langle u,w\rangle + b\langle v,w\rangle + a\langle \hat{u},\hat{w}\rangle + b\langle \hat{v},\hat{w}\rangle \\ &= a\langle (u,\hat{u}),(w,\hat{w})\rangle + b\langle (v,\hat{v}),(w,\hat{w})\rangle.\end{aligned}$$

Since the inner product over H^1 is positive-definite we easily see that the inner product over $\mathbb{V}$ is positive-definite. Hence, $\mathbb{V}$ is an inner product space. It remains then to show that it is complete. Let $(v_n,\hat{v}_n)$ be Cauchy sequences in $\mathbb{V}$ converging to $(v,\hat{v})$, using the norm we then have

$$\|(v_n - v, \hat{v}_n - \hat{v})\|_{\mathbb{V}}^2 = \|v_n - v\|_{H^1(\Omega)}^2 + \|\hat{v}_n - v\|_{H^1(\Lambda)}^2 \to 0$$

which shows that v_n and $\hat{v}_n$ are Cauchy sequences in $H^1(\Omega)$ and $H^1(\Lambda)$ respectively. To show completeness we now need to show that $(v,\hat{v})$ is in $\mathbb{V}$. Again, using the norm, we have

$$\begin{aligned}\|(v_n,\hat{v}_n) - (v,\hat{v})\|_{\mathbb{V}}^2 &= \langle (v_n,\hat{v}_n) - (v,\hat{v})\rangle \\ &= \langle (v_n - v),(v_n - v)\rangle + \langle (v_n - v),(v_n - v)\rangle \\ &= \|v_n - v\|_{H^1(\Omega)}^2 + \|\hat{v}_n - \hat{v}\|_{H^1(\Lambda)}^2 \to 0.\end{aligned}$$

■

Defining

$$B[(u,v),(\hat{u},\hat{v})] := \int_\Omega \nabla u \cdot \nabla v dx + \int_\Lambda \nabla \hat{u} \cdot \nabla \hat{v} ds + \gamma \int_\Lambda (u-v)(\hat{u}-\hat{v}) ds$$

The above theorem ensures that we are in the familiar setting of Theorem 3.1.6, seeking a bilinear operator $B : \mathbb{V} \times \mathbb{V} \to \mathbb{R}$, where $\mathbb{V}$ are Hilbert spaces. The above theorem can also easily be extended to hold for the product of N Hilbert spaces, showing that the Hilbert space defined in (3.41) makes sense. The next steps are to show boundedness and coercivity of B.

Boundedness

We consider the terms in B separately and by the Cauchy–Schwarz inequality we have

$$\left|\int_\Omega \nabla u \cdot \nabla v dx\right| \leq \int_\Omega |\nabla u \cdot \nabla v| dx \leq \|\nabla u\|_{L^2(\Omega)} \|\nabla v\|_{L^2(\Omega)}.$$

Furthermore, since $\|w\|_{H^1}^2 = \|\nabla w\|_{L^2}^2 + \|w\|_{L^2}^2$, the above is bounded by

$$|\int_\Omega \nabla u \cdot \nabla v dx| \leq \|u\|_{H^1(\Omega)} \|v\|_{H^1(\Omega)}. \tag{3.45}$$

and by (3.44) we obtain the desired bound

$$|\int_\Omega \nabla u \cdot \nabla v dx| \leq \|(u, \hat{u})\|_{\mathbb{V}} \|(v, \hat{v})\|_{\mathbb{V}}. \tag{3.46}$$

Similarly, we have

$$|\int_\Lambda \nabla \hat{u} \cdot \nabla \hat{v} ds| \leq \|(u, \hat{u})\|_{\mathbb{V}} \|(v, \hat{v})\|_{\mathbb{V}}.$$

For the last expression in the bilinear form we use the triangle inequality and Cauchy-Schwarz to obtain

$$\begin{aligned} |\gamma \int_\Lambda (u - \hat{u})(v - \hat{v}) ds| &\leq \gamma \int_\Lambda |u - \hat{u}||v - \hat{v}| ds \\ &\leq \gamma \|u - \hat{u}\|_{L^2(\Lambda)} \|v - \hat{v}\|_{L^2(\Lambda)} \\ &\leq \gamma \left(\|u\|_{L^2(\Lambda)} + \|\hat{u}\|_{L^2(\Lambda)} \right) \left(\|v\|_{L^2(\Lambda)} + \|\hat{v}\|_{L^2(\Lambda)} \right). \end{aligned}$$

Recall, that we denote u_Λ as the centerline trace of u, by the trace inequality (3.5) we then obtain

$$\begin{aligned} &|\gamma \int_\Lambda (u - \hat{u})(v - \hat{v}) ds| \leq \gamma (C_T \|u\|_{H^1(\Omega)} + \|\hat{u}\|_{L^2(\Lambda)})(C_T \|v\|_{H^1(\Omega)} + \|\hat{v}\|_{L^2(\Lambda)}) \\ &\leq \gamma (C_T + 1) | \|u\|_{H^1(\Omega)} + \|\hat{u}\|_{H^1(\Lambda)} | (C_T + 1) \|v\|_{H^1(\Omega)} + \|\hat{v}\|_{H^1(\Lambda)}. \end{aligned}$$

Note, that we could have used $\max\{C_T, 1\}$ but for simplicity, we choose $(C_T + 1)$ ensuring that the constant is greater than one. Lastly we use that $\|\mathbf{x}\|_{l^1} \leq \sqrt{d} \|\mathbf{x}\|_{l^2}$, for $\mathbf{x} \in \mathbb{R}^d$ to obtain

$$|\gamma \int_\Lambda (u - \hat{u})(v - \hat{v}) ds| = \gamma (C_T + 1)^2 \sqrt{2} \|(u, \hat{u})\|_{\mathbb{V}} \|(v, \hat{v})\|_{\mathbb{V}}.$$

Summarizing we have

$$\begin{aligned} |B[(u, \hat{u}), (v, \hat{v})] &\leq (\gamma (C_T + 1)^2 \sqrt{2} + 1 + 1) \|(u, \hat{u})\|_{\mathbb{V}} \|(v, \hat{v})\|_{\mathbb{V}} \\ &= (\gamma (C_T + 1)^2 \sqrt{2} + 2) \|(u, \hat{u})\|_{\mathbb{V}} \|(v, \hat{v})\|_{\mathbb{V}} \end{aligned} \tag{3.47}$$

which shows boundedness with constant $\alpha = (\gamma (C_T + 1)^2 \sqrt{2} + 2)$.

Coercivity

We have defined different trial and test spaces for the solutions $(u, \hat{u})$ and test functions $(v, \hat{v})$, however, Theorem 3.1.6 require them to be equal. Poincare type inequalities shows us that trace zero spaces like H_0^1, $H_{\Sigma_0}^1$ are subspaces of H^1 but this is not true for H_D^1 and $H_{\Sigma_D}^1$. To show coercivity we therefore need to reformulate (3.43), seeking also solutions in $H_{\Sigma_0}^1 \times H_0^1(\Lambda)$.

We will use two Poincare-type inequalities, one for the 2D solution u and another for the 1D solution $\hat{u}$. Consider first the latter, we seek a solution

$\hat{u} \in H^1_D(\Lambda) = \{\hat{u} \in H^1(\Lambda) : \hat{u} = \hat{g} \text{ on } \partial\Lambda\}$ and define $\hat{u} = \hat{u}_0 + E\hat{g}$, where $E\hat{g}$ denotes an extension of the boundary data to the whole of Λ. Inserting this in the bilinear operator we obtain

$$\begin{aligned}\int_\Omega \nabla u \cdot \nabla v dx + \int_\Lambda \nabla(\hat{u}_0 + E\hat{g}) \cdot \nabla \hat{v} ds + \gamma \int_\Lambda (u - (\hat{u}_0 + E\hat{g}))(v - \hat{v}) ds \\ = \int_\Omega \nabla u \cdot \nabla v dx + \int_\Lambda \nabla \hat{u}_0 \cdot \nabla \hat{v} ds + \int_\Lambda (u - \hat{u}_0)(v - \hat{v}) ds \\ = \int_\Lambda E\hat{g}(v - \hat{v}) - \nabla E\hat{g} \cdot \nabla \hat{v} ds.\end{aligned}$$

Since $\hat{u}_0$ is zero on the boundary $\partial\Lambda$ we can use the following lemma

Lemma 3.3.2 (Poincaré for H^1_0)**.** *Assume Λ is a bounded open subset of $\mathbb{R}$ and that $u \in H^1_0(\Lambda)$, then the following inequality holds*

$$\|u\|_{L^2(\Omega)} \leq C_p \|\nabla u\|_{L^2(\Omega)} \tag{3.48}$$

Proof. (See e.g. Evans, 1998, p. 280). ■

For the 2D solution u we seek $u \in H^1_{\Sigma_D} = \{u \in H^1(\Omega) : u = g \text{ on } \Sigma\}$. Since Σ is a subset of the boundary $\partial\Omega$ it will require a similar but different treatment as the 1D solution. We define $u = u_0 + Eg$ such that $u_0 \in H^1_{\Sigma_0} = \{H^1(\Omega) : u_0 = 0 \text{ on } \Sigma\}$, the bilinear operator then becomes

$$\begin{aligned}\int_\Omega \nabla(u_0 + Eg) \cdot \nabla v dx + \int_\Lambda \nabla \hat{u}_0 \cdot \nabla \hat{v} ds + \gamma \int_\Lambda ((u_0 + Eg) - \hat{u}_0)(v - \hat{v}) ds \\ = \int_\Omega \nabla u_0 \cdot \nabla v + \int_\Lambda \nabla \hat{u}_0 \cdot \nabla \hat{v} ds + \gamma \int_\Lambda (u_0 - \hat{u}_0)(v - \hat{v}) ds \\ = -\int_\Omega \nabla Eg \cdot \nabla v dx - \int_\Lambda Eg(v - \hat{v}) ds.\end{aligned}$$

Since u_0 has vanishing trace on $\Sigma = \partial\Omega/\Lambda$ we can use the following lemma.

Lemma 3.3.3. *Let Σ be a Lipschitz continuous subset of the boundary $\partial\Omega$ and let $u \in H^1_{\Sigma_0}(\Omega)$ be functions with vanishing trace on Σ, then*

$$\|u\|_{L^2(\Omega)} \leq C_{p\Sigma} \|\nabla u\|_{L^2(\Omega)} \tag{3.49}$$

Proof. The lemma is stated as in (Cerroni et al., 2019) and the proof can be found in (e.g. Salsa, 2016, p. 489). ■

We now have $B[(u_0, \hat{u}_0), (v_0, \hat{v}_0)] : \mathbb{V} \times \mathbb{V} \to \mathbb{R}$, where $\mathbb{V} = H^1_{\Sigma_0}(\Omega) \times H^1_0(\Lambda)$ and the right hand side

$$F(v, \hat{v}) = -\int_\Omega \nabla Eg \cdot \nabla v dx - \int_\Lambda \nabla E\hat{g} \cdot \nabla \hat{v} ds + \int_\Lambda (E\hat{g} - Eg)(v - \hat{v}) ds. \tag{3.50}$$

To show coercivity we want to bound

$$B[(u_0, \hat{u_0}), (u_0, \hat{u}_0)] = \int_\Omega (\nabla u_0)^2 dx + \int_\Lambda (\nabla \hat{u}_0)^2 + \gamma \int_\Lambda (u_0 - \hat{u}_0)^2 ds$$

below and since $\gamma \geq 0$ we have

$$B[(u_0, \hat{u}_0), (u_0, \hat{u}_0)] \geq \int_\Omega (\nabla u_0)^2 dx + \int_\Lambda (\nabla \hat{u}_0)^2 ds.$$

Then, isolating the first term, we have

$$\begin{aligned}\int_\Omega (\nabla u_0)^2 dx = \|\nabla u_0\|^2_{L^2(\Omega)} &= \frac{1}{2}\|\nabla u_0\|^2_{L^2(\Omega)} + \frac{1}{2}\|\nabla u_0\|^2_{L^2(\Omega)} \\ &\geq \frac{1}{2}\|\nabla u_0\|^2_{L^2(\Omega)} + \frac{1}{2C^2_{p\Sigma}}\|u\|^2_{L^2(\Omega)} \\ &\geq \min\{\frac{1}{2}, \frac{1}{2C^2_{p\Sigma}}\}\|u_0\|^2_{H^1(\Omega)}\end{aligned}$$

by Lemma 3.3.3. Similarly, we obtain

$$\int_\Lambda (\nabla \hat{u}_0)^2 \geq \min\{\frac{1}{2}, \frac{1}{2C^2_p}\}\|\hat{u}_0\|^2_{H^1(\Lambda)}$$

by Lemma 3.3.2. Finally, letting $C_1 = \min\{\frac{1}{2}, \frac{1}{2C^2_{p\Sigma}}\}$ and $C_2 = \min\{\frac{1}{2}, \frac{1}{2C^2_p}\}$ the coercivity constant $\beta = \min\{C_1, C_2\}$ provides a lower bound on the bilinear operator

$$B[(u_0, \hat{u_0}), (u_0, \hat{u}_0)] \geq \beta\|(u_0, \hat{u}_0)\|^2_{\mathbb{V}}.$$

Boundedness of F

Lastly, we need to show that $F((v, \hat{v})) : \mathbb{V} \to \mathbb{R}$ is a bounded linear functional on $\mathbb{V}$. We note that the form of F is similar to B which is bounded and thus continuous on $\mathbb{V}$ for $(u, \hat{u}) \in H^1(\Omega) \times H^1(\Lambda)$ [3] hence we require $(Eg, E\hat{g}) \in H^1(\Omega) \times H^1(\Lambda)$. Recalling our definitions on the extension and trace operators and the requirement on the boundary data as functions $g \in H^{\frac{1}{2}}(\Sigma)$, $\hat{g} \in H^{\frac{1}{2}}(\partial\Lambda)$ there does indeed exists a function $Eg \in H^1(\Omega)$ with the trace $Eg|_\Sigma = g$ and $E\hat{g} \in H^1(\Lambda)$ with the trace $E\hat{g}|_{\partial\Lambda} = \hat{g}$.

Summarizing, we have shown that there exists a unique $(u, \hat{u}) \in H^1_{\Sigma_D}(\Omega) \times H^1_D(\Lambda)$ such that

$$\begin{aligned}\int_\Omega \nabla u \cdot \nabla v dx + \int_\Lambda \nabla \hat{u} \cdot \nabla \hat{v} ds + \gamma \int_\Lambda (u - \hat{u})(v - \hat{v}) ds \\ = 0 \ \ \forall (v, \hat{v}) \in H^1_{\Sigma_0}(\Omega) \times H^1_0(\Lambda)\end{aligned} \tag{3.51}$$

via Theorem 3.1.6 and the constants

$$\alpha = (\gamma(C_T + 1)^2\sqrt{2} + 2) \tag{3.52}$$

$$\beta = \min\{C_\Omega, C_\Lambda\} \tag{3.53}$$

representing the upper and lower bound of the bilinear operator.
For ease of notation we from here on denote Dirichlet boundary conditions as

[3] We did not require trace zero to show boundedness of B.

subscripts to the trial space and do not provide how this changes the right-hand side F. To see how the Dirichlet boundary conditions are enforced and specifically how the extensions of the boundary data $Eg, E\hat{g}$ are constructed we refer to (Badia & Verdugo, 2020).

3.4 Galerkin finite element method

Consider an abstract problem on a Hilbert space H, namely, find $w \in H$ such that

$$B[w, z] = F(z) \qquad \forall \ z \in H \tag{3.54}$$

where $B[\cdot, \cdot]$ is a bilinear operator and $F(z)$ is a bounded linear functional on H. By Theorem 3.1.6 presuming coercivity and boundedness of $B[\cdot, \cdot]$, we know that there exists a unique solution $w \in H$ to (3.54). Similarly, we can define a finite approximation w_h to w on a finite-dimensional subspace $H_h \subset H$. Then (3.54) becomes, find $w_h \in H_h$ such that

$$B[w_h, z_h] = F(z_h) \qquad \forall \ z_h \in H_h. \tag{3.55}$$

which is often denoted "the Galerkin equation". Inserting the error $e = w_h - w$ in the above we get

$$B[e, z_h] = B[w_h, z_h] - B[w, z_h] = F(z_h) - F(z_h) = 0 \qquad \forall z_h \in H_h.$$

This implies B-orthogonality of the approximation error e. Assuming that $B[\cdot, \cdot]$ is symmetric, i.e $B[w, z] = B[z, w]$, the bilinear operator defines an inner product in H and induces the norm $\|w\|_B = \sqrt{B[w, w]}$, whence, using the B-orthogonality and the Cauchy-Schwartz inequality we have

$$\begin{aligned} \|w - w_h\|_B^2 &= B[w - w_h, w - w_h] \\ &= B[w - w_h, w] - B[w - w_h, w_h] \\ &= B[w - w_h, w] - B[w - w_h, z_h] \qquad \forall z_h \in H_h \\ &= B[w - w_h, w - z_h] \leq \|w - w_h\|_B \|w - z_h\|_B. \end{aligned}$$

Dividing with $\|u - u_h\|_B$ we obtain the estimate

$$\|w - w_h\|_B \leq \|w - z_h\|_B. \tag{3.56}$$

Since this holds for all $z_h \in H_h$ we know that w_h is the best approximation to w in H_h measured in the B-norm. Motivated by this we want to numerically find an approximation to w by solving (3.55).

In this thesis we will consider problems on the form (3.55) seeking approximations $(u_h, \hat{u}_h) \in V_h \times \hat{V}_h$ to $(u, \hat{u}) \in V \times \hat{V}$. We let $\Omega \in \mathbb{R}^2$ denote the domain restriction to u and Λ the domain restriction to $\hat{u}$. We will always restrict ourselves to the case where $\Lambda \subset \Omega$ i.e. the one-dimensional domain Λ is restricted to live inside the two-dimensional domain Ω. Ω will in fact work as a bounding box of Λ, but further details on this will be provided in chapter 5. With these simple requirements on (Ω, Λ) we briefly describe how we build the approximations $u_h, \hat{u}_h$.
We let $\mathcal{P}_h$ denote an admissible triangulation of (Ω, Λ), then

$$\Omega = \bigcup_{\triangle_k^h \in \mathcal{P}_h} \triangle_k^h \tag{3.57}$$

where h is a refinement parameter and the partitioning of (Ω, Λ) is generated such that the length of all the triangle edges is bound by the choice of h. $\mathcal{P}_h$ consists of triangle elements $\triangle_k^h$ aligned such that

$$\Lambda = \bigcup_{i_k^h \in \mathcal{I}^h} i_k^h \tag{3.58}$$

is made up of a set of triangle edges in $\mathcal{P}_h$. We refer to this as the embedding of the domain Λ in Ω. $\mathcal{I}_h$ is then the admissible partitioning of Λ with line segments i_k^h and thus, h also bounds the length of each line segment i_k^h.
Let $\{\phi_1, \phi_2, ..., \phi_n\}$ be basis functions for V_h consisting of continuous Lagrangian polynomials and similarly let $\{\hat{\phi}_1, \hat{\phi}_2, ..., \hat{\phi}_m\}$ be basis functions for $\hat{V}_h$. Then by letting $\{\boldsymbol{\xi}_i\}_{i=1}^n$ be the nodal values in $\mathcal{P}_h$ and $\{\boldsymbol{\xi}_i\}_{i=1}^m \subset \{\boldsymbol{\xi}_i\}_{i=1}^n$ be the nodal values in $\mathcal{I}_h$ we have

$$\phi_i(\boldsymbol{\xi}_j) = \delta_{ij}, \quad \text{and} \quad \hat{\phi}_i(\boldsymbol{\xi}_j) = \delta_{ij}$$

where δ_{ij} is the Kronecker-Delta function

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j. \end{cases}$$

Since $u_h \in V_h$ and $\hat{u}_h \in \hat{V}_h$ they can be expressed as a linear combination of the basis functions, i.e.

$$u_h = \sum_{i=1}^{n} \mathbf{u}_i \phi_i, \qquad \hat{u}_h = \sum_{i=1}^{n} \hat{\mathbf{u}}_i \hat{\phi}_i \tag{3.59}$$

and similarly for the test functions $v_h, \hat{v}_h$.
Expanding the last term in the previously introduced boundary value problem (3.43) we have

$$\gamma \int_\Lambda (u - \hat{u})(v - \hat{v}) ds = \gamma \int_\Lambda (uv - u\hat{v} - \hat{u}v + \hat{u}\hat{v})\, ds$$

then we let

$$A_{\mathcal{W}}(w_1, w_2) = \int_{\mathcal{W}} \nabla w_1 \cdot \nabla w_2 dw + \gamma \int_\Lambda w_1 w_2 ds \tag{3.60}$$

$$B(w_1, w_1) = \gamma \int_\Lambda w_1 w_2 ds \tag{3.61}$$

such that

$$A_\Omega(u, v) + A_\Lambda(\hat{u}, \hat{v}) - B(u, \hat{v}) - B(\hat{u}, v) = B[(u, \hat{u}), (v, \hat{v})].$$

Considering the terms above and the right hand side of (3.43) we insert our ansatz' (3.59) to obtain

$$\sum_{i=1}^{n} \mathbf{u}_i \left[\int_\Omega \nabla \phi_i \cdot \nabla \phi_j dx + \int_\Lambda \phi_i \phi_j ds \right] + \sum_{i=1}^{m} \hat{\mathbf{u}}_i \left[\gamma \int_\Lambda \hat{\phi}_i \phi_j ds \right] = 0 \qquad \forall \phi_j \in V_h$$

and

$$\sum_{i=1}^{n} \hat{\mathbf{u}}_i \left[\int_\Lambda \nabla \hat{\phi}_i \cdot \nabla \hat{\phi}_j dx + \int_\Lambda \hat{\phi}_i \hat{\phi}_j ds \right] + \sum_{i=1}^{n} \mathbf{u}_i \left[\gamma \int_\Lambda \phi_i \hat{\phi}_j ds \right] = 0 \qquad \forall \hat{\phi}_j \in \hat{V}_h$$

Since the above is a linear system of equations we can express it as

$$\begin{bmatrix} A_\Omega & -B_1 \\ -B_2 & A_\Lambda \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \hat{\mathbf{u}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{3.62}$$

where $A_\Omega^{ij} = A(\phi_i, \phi_j), A_\Lambda^{ij} = A(\hat{\phi}_i, \hat{\phi}_j), B_1^{ij} = B(\hat{\phi}_i, \phi_j)$ and $B_2^{ij} = B(\phi_i, \hat{\phi}_j)$.

We will partition our domain as in (3.57) and (3.58) using the Gmsh (Geuzaine & Remacle, 2009) software and assemble and solve the system (3.62) with the Gridap (Badia & Verdugo, 2020) software. Both first and second-order interpolating polynomials will be considered but we will restrict ourselves to Lagrangian basis functions. The integral calculation needed to obtain the entries of the matrices in (3.62) is done using Gaussian quadrature (see e.g. Quarteroni et al., 2006, Chapter. 10). We note that an n-point Gaussian quadrature rule is exact when integrating polynomials of degree $2n - 1$. Figure 3.3 illustrates the

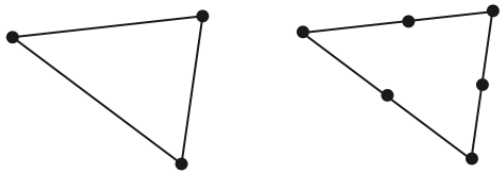


Figure 3.3: Nodal locations in P_1 and P_2 elements.

nodal locations for P_1 and P_2 type elements, where P denotes triangle elements and the subsequent number is the polynomial order of the basis functions $\phi_i, \hat{\phi}_i$. We will also the notation Q_1 and Q_2 for quadrilateral elements.

Error estimates

Theorem 3.1.6 provides existence and uniqueness to the previously introduced boundary value problem (3.51) and by the Galerkin method we can find an approximation to the unique solution $(u, \hat{u})$. A natural question then arises of how the error i.e. the difference between our approximation $(u_h, \hat{u}_h)$ and the solution $(u, \hat{u})$ behaves and to answer this question we want to establish a bound on this difference.
We note that the bilinear operator is symmetric i.e.

$$B[(u, \hat{u}), (v, \hat{v})] = B[(v, \hat{v}), (u, \hat{u})]$$

[4] then, by (3.56) we have

$$\|(u-u_h,\hat{u}-\hat{u}_h)\|_B \leq \|(u-v_h,\hat{u}-\hat{v}_h)\|_B \qquad \forall (v_h,\hat{v}_h)\in \mathbb{V}_h$$

where $\|\cdot\|_B = \sqrt{B[\cdot,\cdot]}$. Using the coercivity and boundedness of $B[\cdot,\cdot]$ we obtain

$$\begin{aligned}\beta\|(u-u_h,\hat{u}-\hat{u}_h)\|_{\mathbb{V}}^2 &\leq B[(u-u_h,\hat{u}-\hat{u}_h),(u-u_h,\hat{u}-\hat{u}_h)]\\ &= \|(u-u_h,\hat{u}-\hat{u}_h)\|_B^2\\ &\leq \|(u-v_h,\hat{u}-\hat{v}_h)\|_B^2\\ &\leq \|(u-v_h,\hat{u}-\hat{v}_h)\|_{\mathbb{V}}^2 \qquad \forall (v_h,\hat{v}_h)\in \mathbb{V}_h.\end{aligned}$$

Hence,

$$\begin{aligned}\|(u-u_h,\hat{u}-\hat{u}_h)\|_{\mathbb{V}}^2 &\leq \frac{\alpha}{\beta}\|(u-v_h,\hat{u}-\hat{v}_h)\|_{\mathbb{V}}^2\\ &\leq \frac{\alpha}{\beta}\left[\|u-v_h\|_{H^1(\Omega)}^2+\|\hat{u}-\hat{v}_h\|_{H^1(\Lambda)}^2\right] \qquad \forall (v_h,\hat{v}_h)\in \mathbb{V}_h.\end{aligned} \tag{3.63}$$

Since this holds for all $(v_h,\hat{v}_h)\in\mathbb{V}_h$ we can choose $(v_h,\hat{v}_h)=(I_h^t u, I_h^t\hat{u})$ i.e. the interpolation of order t of $(u,\hat{u})$ in $\mathbb{V}_h$. Provided sufficient regularity on the triangulation $\mathcal{P}_h$ and using piecewise interpolating polynomials $t=1$ we have from (Braess, 2007, Theorem 6.4) the interpolation error

$$\|u-I_h^1 u\|_{H^1(\Omega)} \leq C_\Omega h\|D^2 u\|_{L^2(\Omega)} \tag{3.64}$$

where D^2u denotes the second order weak or distributive derivative of u and h is a measure on the distance between the nodes in $\mathcal{P}_h$. The above estimate shows that if we can bound D^2u i.e. if u is H^2-regular then we should expect first-order convergence. Similarly, we have from (Cerroni et al., 2019) that the one dimensional interpolation error in $\hat{u}-I_h^1\hat{u}$ can be bounded by

$$\|\hat{u}-I_h^1\hat{u}\|_{H^1(\Lambda)} \leq hC_\Lambda\|D^2\hat{u}\|_{L^2(\Lambda)}. \tag{3.65}$$

Inserting the above estimates in (3.63) we obtain

$$\|(u-u_h,\hat{u}-\hat{u}_h)\|_{\mathbb{V}}^2 \leq \frac{\alpha}{\beta}\left[C_\Omega^2 h_\Omega^2\|D^2u\|_{L^2(\Omega)}^2 + C_\Lambda^2 h_\Lambda^2\|D^2\hat{u}\|_{L^2(\Lambda)}^2\right]. \tag{3.66}$$

Since $\mathcal{I}_h\subset\mathcal{P}_h$ we have $h_\Omega\geq h_\Lambda$ and by taking the scare root of both sides of the equations above we obtain an estimate on the error measured in the $\mathbb{V}$-norm

$$\|(u-u_h,\hat{u}-\hat{u}_h)\|_{\mathbb{V}} \leq h\sqrt{\frac{\alpha(\gamma)}{\beta}}\left[C_\Omega^2\|D^2u\|_{L^2(\Omega)}^2 + C_\Lambda^2\|D^2\hat{u}\|_{L^2(\Lambda)}^2\right]^{\frac{1}{2}} \tag{3.67}$$

where we have set $h=h_\Omega$ and emphasized the dependence of the coupling factor γ by letting the boundedness constant $\alpha=\alpha(\gamma)$.

[4] Because $B[\cdot,\cdot]$ is symmetric a much simpler proof for existence and uniqueness can be obtained using the Riez representation theorem 3.1.4 directly, however since we wanted to obtain the constants α, β we chose the framework of the Lax-Milgram theorem 3.1.6

3.5 Method of manufactured solutions

We want to investigate convergence rates for different solutions to the boundary value problem; find $(u, \hat{u}) \in H^1_{\Sigma_D}(\Omega) \times H^1_D(\Lambda)$ such that

$$B[(u, \hat{u}), (v, \hat{v})] = F(v, \hat{v}) \qquad \forall (v, \hat{v}) \in H^1_{\Sigma_0}(\Omega) \times H^1_0(\Lambda) \tag{3.68}$$

where $B[\cdot, \cdot]$ is the bilinear form corresponding to the variational problem (3.51) introduced in the previous chapter and $F(\cdot, \cdot)$ is a bounded linear functional. We will use to enforce manufactured solutions $(u_m, \hat{u}_m)$. Specifically, we can choose solutions $(u_m, \hat{u}_m)$ and calculate the corresponding $F(\cdot, \cdot)$ such that when we numerically solve (3.68) we can measure the convergence of the computed solution to the true/manufactured solution. We recall that the Gaussian quadrature rule used for numerically computing the integrals associated with (3.68) is exact. The gradients will be provided by analytic derivation of the manufactured solutions. Since the manufactured solutions and the approximations is defined on the same discretization the error is then only by polynomial approximation of the manufactured solutions. We will allways use Dirichlet boundary conditions corresponding to $u_m|_\Sigma$ and $\hat{u}_m|_{\partial\Lambda}$.

We will investigate the finite element solver with three different geometries of Λ. The aim is to understand how different geometries allow for different regularities on the solution $(u, \hat{u})$ and thus different convergence for the solution $(u_h, \hat{u}_h)$ to the manufactured solutions $(u_m, \hat{u}_m)$.

Manufactured solutions on straight line domain and basic estimates

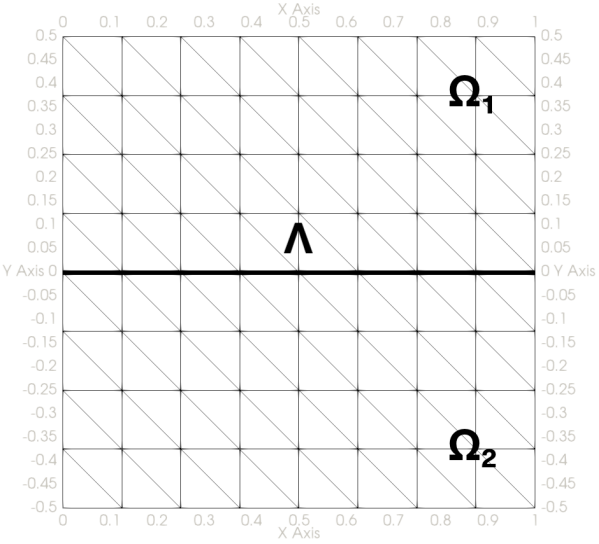


Figure 3.4: Square mesh with straight line embedded interface.

We define Ω as the square with vertices $(0, 0.5), (1, 0.5), (1, -0.5)$ and $(0, -0.5)$ and the subdomains $\Omega_1 = \{(x, y) \in \Omega : y > 0\}$ and $\Omega_2 = \{(x, y) \in \Omega :$

$y < 0\}$. The one dimensional line Λ is embedded in Ω as $\Lambda = \{(x, y) \in \Omega : y = 0\}$. The corresponding strong formulation of (3.68) on the manifold (Ω, Λ) is then

$$\begin{aligned} -\Delta u_1 &= f_1 && \text{on} \quad \Omega_1 \\ -\Delta u_2 &= f_2 && \text{on} \quad \Omega_2 \\ u_1 = u_2 &= u_m && \text{on} \quad \Sigma \end{aligned} \tag{3.69}$$

$$\text{on } \Lambda \quad \begin{cases} -\Delta \hat{u} + \gamma(\hat{u} - u_*) & = \hat{f} \\ \hat{u}|_{\partial\Gamma} & = \hat{u}_m \\ u_1 - u_2 & = 0 \\ -\nabla u_1 \cdot \nu_1 - \nabla u_2 \cdot \nu_2 & = \gamma(u_* - \hat{u}) + g \end{cases} \tag{3.70}$$

where u_* denotes either u_1 and u_2 which are equal on Λ. Recall, $\Sigma = \partial\Omega \setminus \partial\Lambda$. We define $v_1 = v|_{\Omega_1}$, $v_2 = v|_{\Omega_2}$, $u_1 = u|_{\Omega_1}$, $u_2 = u|_{\Omega_2}$ and require continuity by $v_1 - v_2 = u_1 - u_2 = 0$ on Λ. To see how the above equations relate to (3.68) we follow the standard procedure of multiplying with test functions and integrating over the respective domains, using integration by parts and Green's identity we get

$$\left.\begin{aligned} \int_{\Omega_1} \nabla u_1 \cdot \nabla v_1 &= \int_{\Omega_1} f_1 v_1 + \int_{\partial\Omega_1} (\nabla u_1 \cdot \nu_1) v_1 \\ \int_{\Omega_2} \nabla u_2 \cdot \nabla v_2 &= \int_{\Omega_2} f_2 v_2 + \int_{\partial\Omega_2} (\nabla u_2 \cdot \nu_2) v_2 \end{aligned}\right\} \rightarrow$$

$$\begin{aligned} \rightarrow \int_\Omega \nabla u \cdot \nabla v &= \int_\Omega f v - \int_\Lambda (\nabla u_1 \cdot \nu_1 + \nabla u_2 \cdot \nu_2) v \\ &= \int_\Omega f v + \int_\Lambda \gamma(u - \hat{u}) v + g v \end{aligned} \tag{3.71}$$

where we have inserted g from (3.70) and set $v = 0$ on Σ such that the boundary term vanishes on Σ and the remainder is the interface conditions over Λ. Similarly, we get

$$\int_\Lambda \nabla \hat{u} \cdot \nabla \hat{v} + \gamma(\hat{u} - u)\hat{v} = \int_\Lambda \hat{f} \hat{v}. \tag{3.72}$$

Note that since Λ is one dimensional, there does not appear a Neumann term and for the gradient on Λ we have $\nabla = \frac{d}{dx}$, we do however keep the notation ∇ for consistency. Combining (3.71) and (3.72) we obtain the familiar variational formulation; Find $(u, \hat{u}) \in H^1_{\Sigma_D}(\Omega) \times H^1_D(\Lambda)$ s.t.

$$\int_\Omega \nabla u \cdot \nabla v + \int_\Lambda \nabla \hat{u} \cdot \nabla \hat{v} + \int_\Lambda \gamma(u - \hat{u})(v - \hat{v}) = \int_\Omega f v + \int_\Lambda \hat{f} \hat{v} - g v \quad \forall (v, \hat{v}) \in H^1_{\Sigma_0}(\Omega) \times H^1_0(\Lambda) \tag{3.73}$$

where the $H^1_{\Sigma_D}(\Omega) = \{u \in H^1(\Omega) : u = u_m \text{ on } \partial\Sigma\}$ and $H^1_D(\Lambda) = \{\hat{u} \in H^1(\Lambda) : u = \hat{u}_m \text{ on } \partial\Lambda\}$. Using the strong formulation we can construct manufactured solutions by choosing $u = u_m$, $\hat{u} = \hat{u}_m$ and calculating the

corresponding $f, \hat{f}$ and g. With these source terms, our solutions $(u_h, \hat{u}_h)$ to the variational problem above should approximate the manufactured solutions $(u_m, \hat{u}_m)$.

In our basic estimates, we want to control that the solver can approximate the manufactured solutions exactly when the order of our interpolating polynomials and the manufactured solutions agree. We also want to show that the solver behaves as expected for no coupling ($\gamma = 0$) and weak coupling ($\gamma = 1$). The error is defined as the difference of $u_h, \hat{u}_h$ and $u, \hat{u}$ measured in the H^1-norm respectively and combined with the $\mathbb{V}$-norm

$$e_{tot} := \sqrt{\|u - u_h\|^2_{H^1(\Omega)} + \|\hat{u} - \hat{u}_h\|^2_{H^1(\Lambda)}}.$$

	$\gamma = 0$		$\gamma = 1$	
Manufactured Solutions	H^1-error	e_{tot}	H^1-error	e_{tot}
$u_m = x - y$	2.9e-15	2.9e-15	2.2e-15	3.5e-15
$\hat{u}_m = x$	4.1e-15		2.7e-16	
$u_m = x - y$	2.9e-15	0.072	7.3e-4	0.072
$\hat{u}_m = x^2$	0.072		0.072	
$u_m = \cos(x) + \sin(y)$	0.032	0.037	0.032	0.037
$\hat{u}_m = \sin(x)$	0.019		0.019	

Table 3.1: Error in approximating manufactured solutions for the boundary value problem (3.73)

Table 3.1 shows the error for different manufactured solutions. Since we have used $P1$ elements in our approximations we should expect machine precision for manufactured solutions of polynomial degree $t = 1$. The table shows that this is the case and we also note that the approximation error inflicts on u_m when the coupling factor $\gamma \neq 0$ and $\hat{u}_m$ is a higher order polynomial, even though u is a first-order polynomial.
Recall our estimate

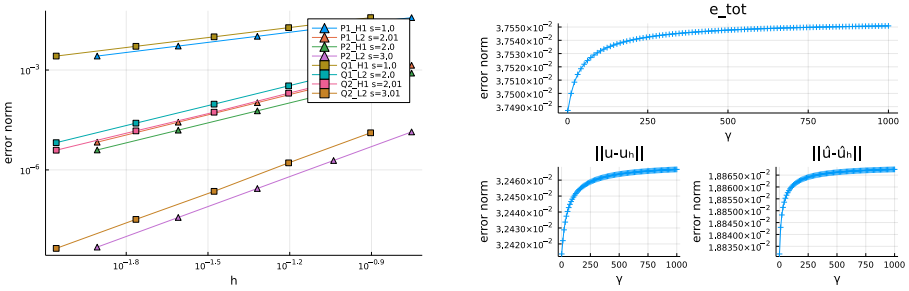
$$e_{tot} = \|(u - u_h, \hat{u} - \hat{u}_h)\|_{\mathbb{V}} \leq h\sqrt{\frac{\alpha(\gamma)}{\beta}} \left[C^2_\Omega \|D^2 u\|^2_{L^2(\Omega)} + C^2_\Lambda \|D^2 \hat{u}\|^2_{L^2(\Lambda)}\right]^{\frac{1}{2}} \tag{3.74}$$

calculating e_{tot} for five successive mesh refinements, approximately halving h for each refinement we should, from the estimate above expect h^1-convergence - for H^2 regular solutions. Furthermore, (Braess, 2007) establishes the more general bound by

$$\|u - I_h u\|_{H^m(\Omega)} \leq C h^{t-m} \|D^t u\|_{L^2(\Omega)} \tag{3.75}$$

for triangle elements when interpolating the 2D solution using $t - 1$ order polynomials and also shows that the bound holds for quadrilateral elements when $t \in [2, 3]$. Although, our estimate (3.74) does not bound the error in the L^2 norm we want to see if (3.75) holds and thus measure the error in both norms (L^2 and H^1) and use both triangular P and quadratic Q elements. Furthermore, provided enough regularity it is natural to suspect that the one-dimensional

approximation also follows the above bound. We thus suspect that e_{tot} follows (3.75) for interpolating polynomials of order 2.
Using the last manufactured solutions in table (3.1) which is clearly both H^2 and H^3 regular we measure the error for four successive refinements on h. From (3.74) and the derivation of the boundedness constant α we see that the error also depends on the coupling factor γ and we should expect an increase in the error if γ increases.



(a) Log-log plot of error measured in both the L^2 -norm and the H^1-norm using both P_t and Q_t, $t \in [1, 2]$ elements.

(b) error as a function of increasing coupling γ using P_1 elements.

Figure 3.5: Convergence analysis of the approximation $(u_h, \hat{u}_h)$ to manufactured solutions $(u_m, \hat{u}_m)$.

Figure 3.5 (a) shows that we obtain the expected convergence rates for both triangular and quadrilateral elements. From Figure 3.5 (b) we see that the error associated with an increase in γ is minuscule as $\Delta e = 6.0 \cdot 10^{-5}$ while $\gamma \in [0, 1000]$. For the rest of our study on manufactured solutions we therefore simply set $\gamma = 1$. Further experiments providing intuition on the specific shape of the graphs in Figure 3.5 (b) are provided in Appendix A.

The boundary value problem (3.73) admits solution $(u, \hat{u}) \in H^1 \notin H^2$, however from the estimate (3.74) we see that that the regularity of the solution clearly affects the bound on the error. To study convergence for lower regularity of the solution we let $u_m = \sin|y|$ or equivalently

$$u_m = \begin{cases} \sin(y) & \text{on } \Omega_1 \\ -\sin(y) & \text{on } \Omega_2 \end{cases} \tag{3.76}$$

and, for simplicity, we set $\hat{u} = 0$ on Λ. We then have

$$u_m(x, y) = \sin|y|, \qquad \frac{d}{dy} u_m(x, y) = \frac{y}{|y|}\cos|y| = \operatorname{sgn}(y)\cos|y|$$
$$\frac{d^2}{dy^2} u_m(x, y) = \frac{d}{dy}\operatorname{sgn}(y)\cos|y| + \operatorname{sgn}(y)\frac{d}{dy}\cos|x|$$

since $\cos(y)$ is symmetric we have $\cos|y| = \cos(y)$ and we can rewrite $\operatorname{sgn}(y) = 2H(y) - 1$ where $H(y)$ is the Heaviside function with the known distributional derivative $\delta(y)$, namely the Dirac-delta function. Whence, we get

$$\frac{d^2}{dy^2} u_m(x, y) = 2\delta(y) - \operatorname{sgn}(y)\sin(y)$$

where we have simplified $2\delta(y)\cos(y) = 2\delta(y)$ since $\cos(0) = 1$ and $\delta(y) = 0$ when $y \neq 0$. The corresponding source terms to our manufactured solutions are then

$$\begin{aligned} f_1 &= -\sin(y), \qquad f_2 = \sin(y), \qquad \hat{f} = 0 \\ g &= -(0, \cos(y)) \cdot (0,1) - (0, -\cos(y)) \cdot (0,-1) - \gamma \sin(|y|) \\ &= -2\cos(y) - \gamma \sin(|y|). \end{aligned}$$

Now, considering the regularity of $u(x,y) = u(y)$ on Ω we want to show, for H^1-regularity, that u has a weak derivative Du and that $\|Du\|_{L^2(\Omega)} < \infty$. Hence, we need to find a measurable function $Du : \Omega \to \mathbb{R}$ such that

$$\int_\Omega u(y)\phi'(y) = -\int_\Omega Du(y)\phi(y)$$

holds for all smooth functions $\phi : \Omega \to \mathbb{R}$ with compact support in Ω. Using integration by parts we have

$$\begin{aligned} \int_\Omega \sin|y|\phi'(y) &= \int_{\Omega_1} \sin(y) - \int_{\Omega_2} \sin(y) \\ &= \big[\sin(y)\phi(y)\big]_{\Omega_1} - \int_{\Omega_1} \cos(y)\phi(y) - \big[\sin(y)\phi(y)\big]_{\Omega_2} + \int_{\Omega_2} \cos(y)\phi(y), \end{aligned}$$

since ϕ has compact support and since $\sin(0) = 0$ we get

$$\int_\Omega \sin|y|\phi'(y) = -\int_{\Omega_1} \cos(y)\phi(y) + \int_{\Omega_2} \cos(y)\phi(y) = -\int_\Omega \mathrm{sgn}(y)\cos(y)\phi(y).$$

Thus $Du(y) = \mathrm{sgn}(y)\cos(y) = \nabla u(y)$.
Next, we check

$$\|Du\|^2_{L^2(\Omega} = \int_\Omega |\mathrm{sgn}(y)\cos(y)| = \int_\Omega |\cos(y)| < \infty$$

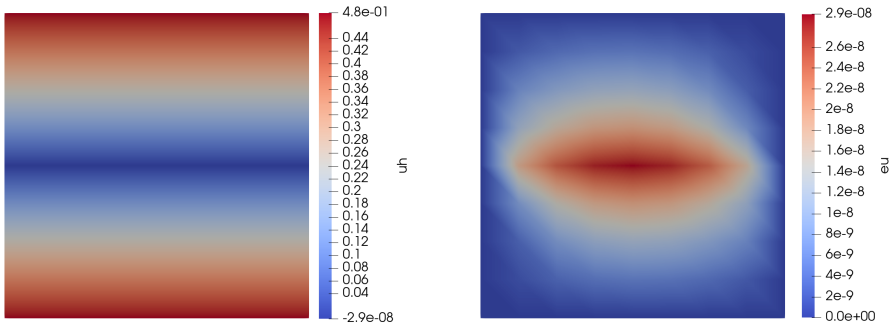
which concludes $u \in H^1(\Omega)$.
Considering the second derivative of u on the two subdomains Ω_1, Ω_2 we have

$$u''(y) = \begin{cases} -\sin(y) & \text{on } \Omega_1 \\ \sin(y) & \text{on } \Omega_2 \end{cases}$$

which we know have to agree with the weak derivative on the domains. Again, using integration by parts we obtain

$$\begin{aligned} \int_\Omega u'\phi' &= \int_{\Omega_1} \cos(y)\phi'(y) - \int_{\Omega_2} \cos(y)\phi'(y) = \big[\cos(y)\phi(y)\big]_{\Omega_1} \\ &\quad + \int_{\Omega_1} \sin(y)\phi(y) - \big[\cos(y)\phi(y)\big]_{\Omega_2} - \int_{\Omega_2} \sin(y)\phi(y) \\ &= (0 - \phi(0)) - (\phi(0) - 0) + \int_\Omega \mathrm{sgn}(y)\sin(y)\phi(y) \\ &= -2\phi(0) + \int_\Omega \mathrm{sgn}(y)\sin(y)\phi(y) \underbrace{\overset{!}{=}}_{\text{should equal}} -\int_\Omega u''(y)\phi(y) = \int_\Omega \mathrm{sgn}(y)\sin(y)\phi(y) \end{aligned}$$

which should hold for all $\phi \in C_c^\infty(\Omega)$ but only holds when $\phi(0) = 0$. Therefore D^2u does not exist and $u \notin H^2(\Omega)$. We do, however, note that the disagreement lies on Λ and that we do have $u \in H^2(\Omega_i)$ for $i = 1, 2$ but without a bound on $\|D^2\hat{u}\|$ we cannot guarantee first order convergence of e_{tot} when refining the mesh parameter h.



(a) Computed solution (b) Error

Figure 3.6: Computed solution u_h and corresponding error $eu = u_m - u_h$.

Figure 3.6 shows the computed solution u_h over the domain $\Omega = \Omega_1 \cup \Omega_2$ and the corresponding error $eu = u - u_h$ for $h \approx 0.17$. We see that the highest error is located on and around Λ. From figure (a), the computed solution u_h on Λ is $-2.9 \cdot 10^{-8}$, while we know that the manufactured solution $u_m = 0$ on Λ, this amounts for the error in (b).

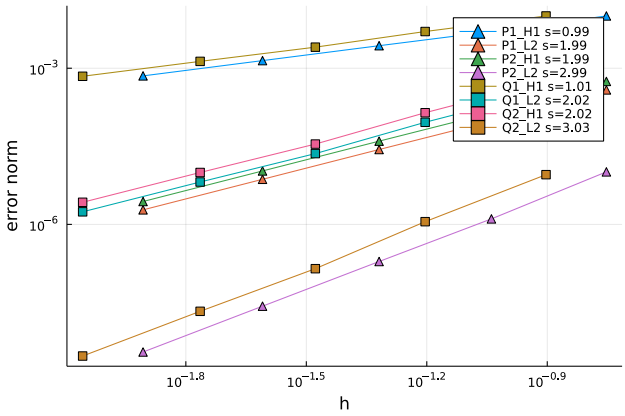


Figure 3.7: Log-log plot of error against mesh refinement for manufactured solutions 3.76

Figure 3.7 shows that the convergence is close to not affected by the lower regularity on u. The calculated slope is 0.994 while with H^2 regularity on the whole of the domain Ω we should expect 1.0. With higher refinement of h or by considering a part of the domain closer to the error in Figure 3.6 (b) we could expect a further decrease in convergence rate.

Nonconvex subdomain

Consider the boundary value problem (3.73) on the square domain $\Omega = \Omega_1 \cup \Omega_2$ with vertices $(-1,1),(1,1),(1,-1)$ and $(-1,-1)$. We define $\Omega_2 = \{(x,y) \in \Omega : (x,y) < 0\}$, $\Omega_1 = \{(x,y) \in \Omega : (x,y) \notin \Omega_2\}$ and $\Lambda = \{(x,y) \in \Omega : (x,y) \in \Omega_1 \cap \Omega_2\}$.

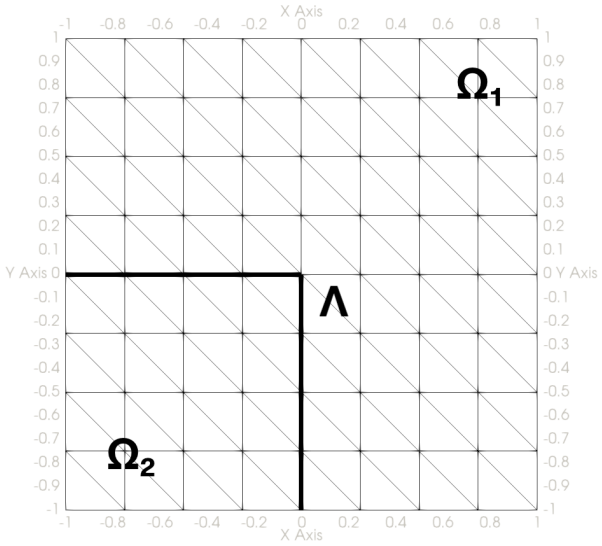


Figure 3.8: Meshed 2D-1D domain, with nonconvex subdomain Ω_1

The reason we want to consider a non-convex domain Ω_1 is that convexity is a well-known requirement on elliptic problems to ensure H^2-regularity of the solution u, (see e.g. Grisvard, 2011, Chapter 2). We note that the corresponding strong formulation of the variational problem (3.73) equal (3.69) and (3.70) but with different definitions on the unit vectors ν_1 and ν_2, this will be emphasised later. We follow our standard procedure of constructing manufactured solutions, but we must pay extra attention to how we implement the solution in Λ. Since $\Lambda = \Lambda_1 \cup \Lambda_2$ where $\Lambda_1 = \{(x,y) \in \Lambda : y = 0\}$ and $\Lambda_2 = \{(x,y) \in \Lambda : x = 0\}$ we will implement the manufactured solution $\hat{u}_m$ piecewise as

$$\hat{u}_m(x,y) = \begin{cases} \hat{u}_1 & (x,y) \in \Lambda_1 \\ \hat{u}_2 & (x,y) \in \Lambda_2. \end{cases}$$

Let us consider solutions $\hat{u}_1$ and $\hat{u}_2$ that are $C^k(\Lambda_i), i \in [1,2]$ with $k \geq 2$. Integration by parts then gives us

$$\begin{aligned} \int_\Lambda \nabla \hat{u}_m \phi' &= \int_{\Lambda_1} \partial_x \hat{u}_1 \phi' + \int_{\Lambda_2} \partial_y \hat{u}_2 \phi' \\ &= [\partial_x \hat{u}_1 \phi]_{\partial\Lambda_1} + [\partial_y \hat{u}_2 \phi]_{\partial\Lambda_2} - \int_{\Lambda_1} \partial_{xx} \hat{u}_1 \phi - \int_{\Lambda_2} \partial_{yy} \hat{u}_2 \phi \\ &= [(\partial_x \hat{u}_1(x,y) - \partial_y \hat{u}_2(x,y))\phi(x,y)]_{(0,0)} - \int_\Lambda \Delta \hat{u} \phi. \end{aligned}$$

As previously, $D^2\hat{u}_m \stackrel{!}{=} \Delta\hat{u}_m$ to obtain $\hat{u}_m \in H^2(\Gamma)$, therefore we require continuity of the derivatives in $(0,0)$ i.e. $\partial_x\hat{u}_1(0,0) = \partial_y\hat{u}_2(0,0)$ or equivalently $\nabla\hat{u}_m \in C^0(\Lambda)$ to ensure $\hat{u}_m \in H^2(\Gamma)$. Notice also that this requirement corresponds to the balance of fluxes in section 3.2. Experiments show that with the above satisfied and thus $u, \hat{u} \in H^2$ we get the expected convergence rates for manufactured solutions $(u_m, \hat{u}_m) \in H^2(\Omega) \times H^2(\Gamma)$.

One reason that higher regularity of the solution to a boundary value problem and convexity of the domain often imply each other is that non-convex domains, like Ω_1 and a particular boundary value problem, can have underlying singularities. (Elman et al., 2014) investigates the solution to the Poisson boundary value problem $-\Delta u = 1$ on a domain similar to Ω_1 with boundary conditions $u = 0$ on $\partial\Omega_1$. They show that the solution u can be closely approximated at the origin by

$$u = r^{2/3}\sin((2\theta + \pi)/3) \tag{3.77}$$

where r represents the radial distance to the centre of Figure 3.8 and θ the angel with the vertical axis y, notice that the radial derivatives of u are unbounded at the origin. Setting

$$u_1(r,\theta) = r^{2/3}\sin((2\theta + \pi)/3) \quad \text{on} \quad \Omega_1 \tag{3.78}$$

$$u_2 = 0 \quad \text{on} \quad \Omega_2 \tag{3.79}$$

$$\hat{u} = 0 \quad \text{on} \quad \Lambda \tag{3.80}$$

as manufactured solutions, we can create a similar scenario as the Poisson boundary value problem on a non-convex domain. We use the definitions

$$r(x,y) := \sqrt{x^2 + y^2} \tag{3.81}$$

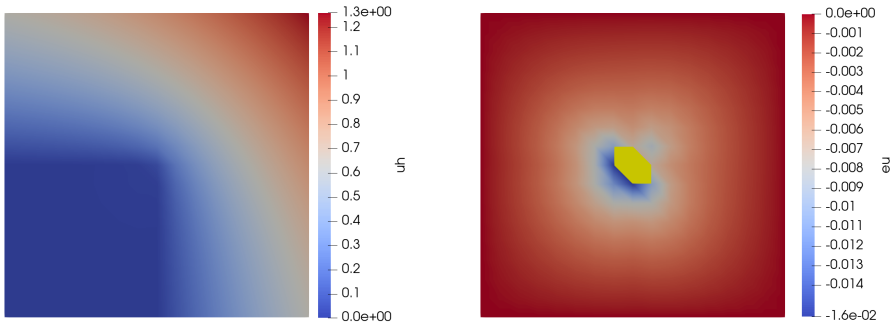
$$\theta(x,y) := \begin{cases} \cos^{-1}(\frac{x}{r}) & y \geq 0 \\ -\cos^{-1}(\frac{x}{r}) & y < 0 \end{cases} \tag{3.82}$$

and note that $\theta(0, y < 0) = -\frac{\pi}{2} \to u_1 = 0$ and $\theta(x < 0, 0) = -1 \to u_1 = 0$ such that the continuity condition $u_1 - u_2 = 0$ on Λ is fulfilled. The condition given last in (3.70) is fulfilled by defining ν_1 as the piecewise unit normal vector (in polar coordinates) from Ω_1 over Λ_1 and Λ_2. ν_2 is defined similarly, but opposite, from Ω_2 to Ω_1.

From Figure 3.9(a) we see that the computed solution cannot approximate the singularity at the origin, and this amounts to the error in Figure 3.9(b). The yellow colour denotes that the value at the origin node is a NaN value and since the plot interpolates the nodal values inside the triangle elements, each triangle sharing the origin node is yellow. For this boundary value problem, $u \in H^1(\Omega) \notin H^2(\Omega)$ and we cannot bound the error by (3.74). However, (Elman et al., 2014) establishes, for the Poisson boundary value problem on an L-shaped domain, the bound

$$\|\nabla(u - u_h)\|_{L^2(\Omega)} \leq Ch^{\frac{2}{3}-\epsilon} \tag{3.83}$$

where $\epsilon > 0$ is an arbitrary constant.



(a) Computed solution (b) Error

Figure 3.9: Computed solution and corresponding error, yellow colour indicated singularity at origin (x,y) = (0,0).

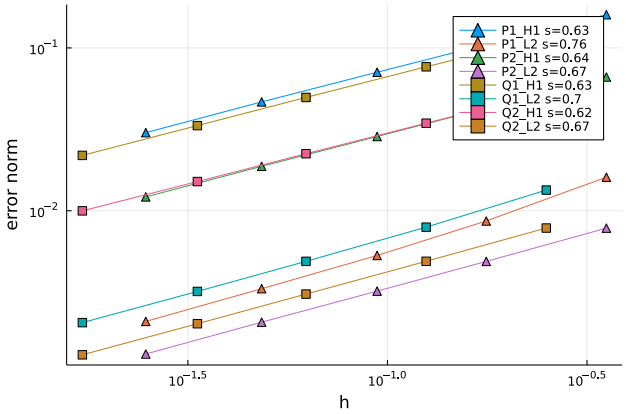


Figure 3.10: Log-log plot of error against mesh refinement.

Figure 3.10 shows that the bound (3.83) holds for four successive refinements on h and for P_1, P_2, Q_1, Q_2 type elements.

Unstructured mesh

Previously we have only considered structured meshes, which we have shown provide regular and uniform errors excluding the point of singularity in Figure 3.9. However, considering a more irregular domain Λ, we are forced to use a less structured mesh to be able to embed $\Lambda \subset \Omega$. Recalling estimate (3.74) we know that the bound on the error is dependent on the length between the nodes h. Considering the 2D triangulation $\mathcal{P}_h$, consisting of triangle elements Δ_k we should expect a different error bound depending on the length of the longest triangle edge h_k. Indeed, following (Elman et al., 2014) we can establish the bound

$$\|D(u - u_h)^2_{\Delta_k}\| \leq 2\frac{h_k^2}{|\Delta_k|}\|D(\bar{u} - \bar{u}_h)\|^2_{\Delta_*}. \tag{3.84}$$

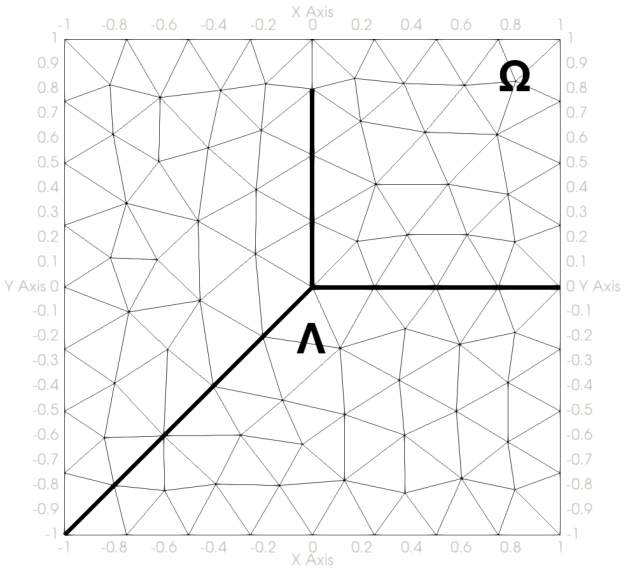


Figure 3.11: Branching domain Λ embedded in 2D domain Ω.

where $\bar{u}$ denotes the mapped function on a reference element Δ_*. (3.84) shows that a large aspect ratio $\frac{h_k^2}{|\Delta_k|}$, i.e. long triangle edges w.r.t the area of the triangle correspond to a high bound associated with that element error, where the lowest bound is obtained for equilateral elements.

We consider the 2D-1D domain given in Figure 3.11 and manufactured solutions

$$u_m = x^3 - y^3 \qquad \text{on } \Omega \tag{3.85}$$

$$\hat{u}_m = x^3 + y^3 \qquad \text{on } \Lambda. \tag{3.86}$$

Since $u_m \in C^2(\Omega)$, the strong formulation of (3.68) is simplified to

$$\begin{aligned} -\Delta u &= f && \text{on } \Omega \\ u &= u_m && \text{on } \Sigma \end{aligned} \tag{3.87}$$

$$\text{on } \Gamma \quad \begin{cases} -\Delta\hat{u} + \gamma(\hat{u} - u) & = \hat{f} \\ \hat{u}|_{\partial\Gamma} & = \hat{u}_m \\ 0 & = \gamma(u - \hat{u}) + g \end{cases}. \tag{3.88}$$

Note that the Dirichlet boundary condition $\hat{u}|_{\partial\Gamma} = \hat{u}_m$ is now enforced on the starting point $(x, y) = (-1, -1)$ and at both ends $(1, 0)$ and $(0, 0.8)$. Using P_1, P_2[5] type elements we will experience an interpolation error in our solution

[5]This curve Λ does not allow for discretization using Qk type elements. Gmsh (Geuzaine & Remacle, 2009) allows for combined element types, but this functionality is not yet implemented in Gridap (Badia & Verdugo, 2020), and will thus not be investigated.

$(u_h, \hat{u}_h)$ but we want to ensure convergence with the more complex Λ and as a consequence, the more irregular mesh. We enforce the manufactured solutions by setting the source terms

$$f = -\Delta u \tag{3.89}$$
$$g = -\gamma(u - \hat{u}) \tag{3.90}$$

and to enforce $\hat{u}_m$ on Λ we divide Λ into

$$\Lambda_{xy} = \begin{cases} \Lambda & x, y \in (-1, 0) \\ \emptyset & \text{else} \end{cases} \qquad \Lambda_x = \begin{cases} \Lambda & x \in (0, 1), y = 0 \\ \emptyset & \text{else} \end{cases}$$
$$\Lambda_y = \begin{cases} \Lambda & y \in (0, 1), x = 0 \\ \emptyset & \text{else.} \end{cases}$$

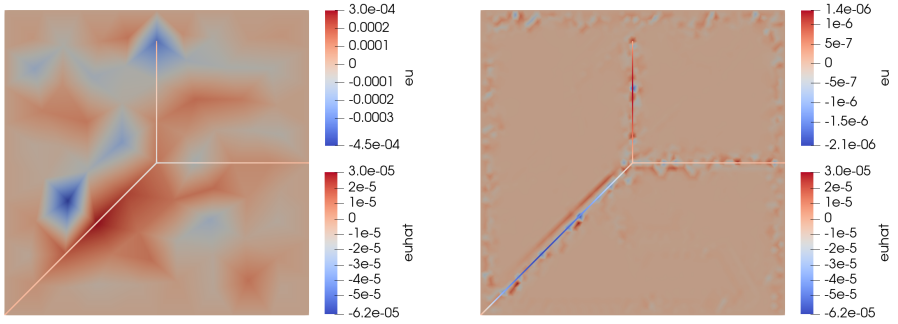
Now, we can separate the different gradients given as

$$\nabla_{\Lambda_{xy}} = \frac{1}{\sqrt{2}}\left(\frac{\partial}{\partial x}, \frac{\partial}{\partial x}\right), \qquad \nabla_{\Lambda_x} = \frac{\partial}{\partial x}, \qquad \nabla_{\Lambda_y} = \frac{\partial}{\partial y}$$

and calculate the different source terms

$$\hat{f}_1 = -\nabla_{\Lambda_{xy}} \cdot \nabla_{\Lambda_{xy}} u = \frac{1}{2}\left(\frac{\partial^2 \hat{u}_m}{\partial x^2} + \frac{\partial^2 \hat{u}_m}{\partial y^2}\right), \qquad \hat{f}_2 = \nabla_{\Lambda_x} \cdot \nabla_{\Lambda_x} = \frac{\partial^2 \hat{u}_m}{\partial x^2}$$
$$\hat{f}_3 = \nabla_{\Lambda_y} \cdot \nabla_{\Lambda_y} = \frac{\partial^2 \hat{u}_m}{\partial y^2}$$

Notice that the $\nabla_{\Lambda_{xy}} \hat{u} = \nabla_{\Lambda_x} \hat{u} = \nabla_{\Lambda_y} \hat{u}$ at the branch point $(0, 0)$ so $\hat{u} \in C^1(\Lambda)$ and thus from our reasoning in the non-convex domain, $\hat{u} \in H^2(\Lambda)$. Figure 3.12



(a) Error on coarsest mesh. (b) Error in finest mesh.

Figure 3.12: Error $u - u_h$ with mesh refinement $h_k \approx 0.35 \to 0.02$.

shows that on an unstructured grid, the location of the interpolation error is decided by the aspect ratio of the triangle elements. The nodes shared between triangle elements with a high aspect ratio will exhibit the highest error. Refining the mesh in Figure 3.12 (b) allows for more equilateral triangle elements but retains the more "bad" elements along the boundary $\partial\Omega$ and Λ. The domain Λ is partitioned into equal distance nodes, therefore the error on Λ is more

regular and the error is concentrated away from the boundary points $\partial\Lambda$ where we enforce $\hat{u}_h = \hat{u}_m$ as Dirichlet boundary conditions. From estimate (3.84) we see that the bound is dependent on the aspect ratio of the triangle elements, in particular, we must prevent that $|\Delta_k| \to 0$ as we refine the mesh. Gmsh (Geuzaine & Remacle, 2009) does this automatically by scaling the elements size and then re-meshing the surface (Ω, Λ). Gmsh also provides a measure of the aspect ratio of the computed triangle elements, using a metric defined by

$$\eta := 2R_1/R_2$$

where R_1 is the inscribed radius and R_2 circumscribed radius. By the relations

$$R_1 = \sqrt{\frac{(s-a)(s-b)(s-c)}{s}}, \qquad R_2 = \frac{abc}{4R_1 s}$$

where a, b, c are the triangle side lengths and $s = \frac{1}{2}(a+b+c)$ is the semi-perimeter. Additionally, we have the formula

$$|\Delta_k| = \sqrt{s(s-a)(s-b)(s-c)}$$

which is often denotes as Heron's formula. Combining the above we can express η by

$$\eta = \frac{2R_1 \cdot 4R_1 s}{abc} = \frac{8R_1^2 s}{abc} = \frac{8|\Delta_k|^2}{sabc} = \frac{16|\Delta_k|^2}{a^2bc + b^2ac + c^2ab}$$

and choosing a, b or c as our longest triangle edge h_k we see that increasing $\frac{h_k^2}{|\Delta_k|}$ i.e. minimizing the triangle area or increasing h_k has the effect of decreasing η. We also note that for an equilateral triangle, we have the best possible aspect ratio and $\eta = 1$. Table 3.2 shows that the aspect ratio does not increase (η

scale	element id	h_{max}	η
1.0	71	0.3504	0.767
0.5	97	0.1572	0.823
0.25	395	0.0773	0.686
0.125	628	0.0437	0.787
0.0625	1158	0.02136	0.737

Table 3.2: Worst elements according to η and corresponding max edge length for five successive mesh refinements.

decrease) as we refine the mesh and that h_{max} is approximately halved for each scaling factor.

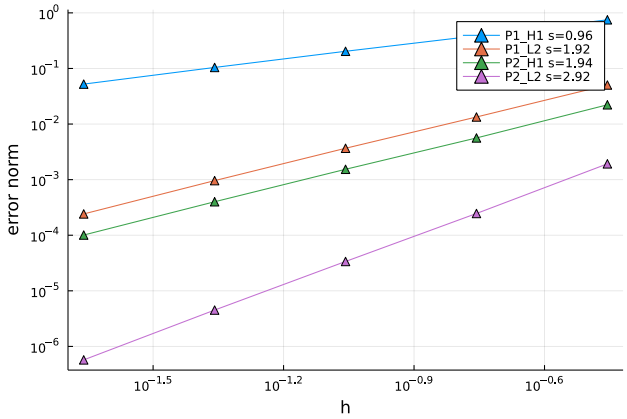


Figure 3.13: Log-log plot of error against mesh refinement.

Figure 3.13 shows that we obtain close to the expected converge rate, but the more varying quality of the triangle elements to some degree slows down the convergence.

3.6 Deciding boundary conditions and error analysis with CCO curvature

We want to consider solutions to the boundary value problem introduced in section 3.2 using vasculature obtained from the method of CCO. We let Ω be the unit square $(0,1)^2$ and Λ the 1D representation of the vasculature and confined in Ω. We chose homogeneous Neumann boundary conditions on Σ and let the Dirichlet boundary conditions on $\partial\Lambda$ be given as

$$D = \frac{1}{1+d_i} \tag{3.91}$$

where d_i is the Euclidean distance from the root proximal point $\mathbf{x}_0$ to a specific distal terminal point, following the path along Λ.

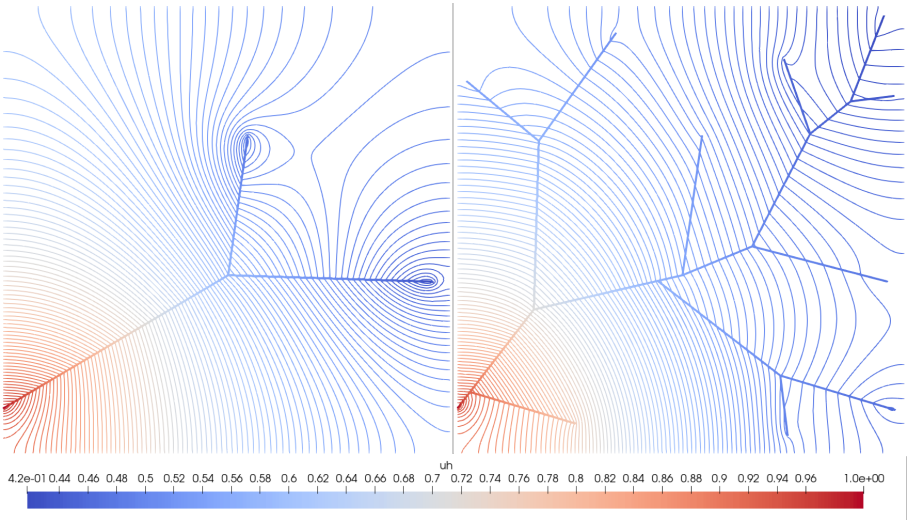


Figure 3.14: Solutions of the boundary value problem with strong coupling $\gamma = 100$ and structures of Λ obtained by the method of CCO.

Figure 3.14 shows the solution u_h as contour lines in the square Ω and $\hat{u}_h$ as a color map on the curvature Λ. Since u_h is purely driven by diffusion and exchange by the high coupling ($\gamma = 100$) with $\hat{u}_h$ we see that the boundary values of $\hat{u}_h$ on the root proximal point and terminal distal points inflicts on the solution u_h surrounding $\hat{u}_h$. The physical motivation for the proposed Dirichlet boundary conditions on $\hat{u}_h$ is then that an injected fluid in the root proximal point will travel along the curvature to the terminal distal points extracting the flow. Since the required energy to "carry" the fluid increases with the length of the path along Λ, we set a difference in pressure that increases with the curve Λ. The homogeneous Neumann boundary conditions are identified in Figure 3.14 by the fact that u_h is constant, normal to the boundary $\partial\Omega$ which indeed implies that the contour lines are perpendicular to $\partial\Omega$. Physically, we motivate this choice of boundary condition by the consideration of only a "patch" of the tissue, in this scenario there would not be a blockage or leakage in the restriction of this patch, but a constant continuation across the boundary.

The method of CCO aims to spread the terminal distal points as homogeneously as possible in Ω and the physical intuition of this is that the

tissue needs to be provided with nutrients by the blood. Considering a uniform piece of tissue the vasculature providing these nutrients should be evenly spread throughout the tissue. The proposed boundary value problem also follows this intuition and hence a consequence of more terminal segments in the vasculature Λ leads to the pressure difference being more evenly spread out in Ω.

Considering an arbitrary number of terminal segments, the complexity of Λ no longer allows for manufactured solutions and we cannot measure convergence as done previously. Indeed our best estimate for the true solution $(u, \hat{u})$ is the solution obtained with high refinement h and since the solution $(u_h, \hat{u}_h)$ should converge to this solution we measure the relative error between successive mesh refinements. We let $\mathbf{u} = (u, \hat{u})$, $h_k = h/2^k$ and measure the relative error by

$$RE(\mathbf{u}_{h_{k+1}}, \mathbf{u}_{h_k}) = \frac{\|\mathbf{u}_{h_{k+1}} - \mathbf{u}_{h_k}\|}{\|\mathbf{u}_{h_{k+1}}\|} \tag{3.92}$$

for both $P1$ and $P2$ type elements.

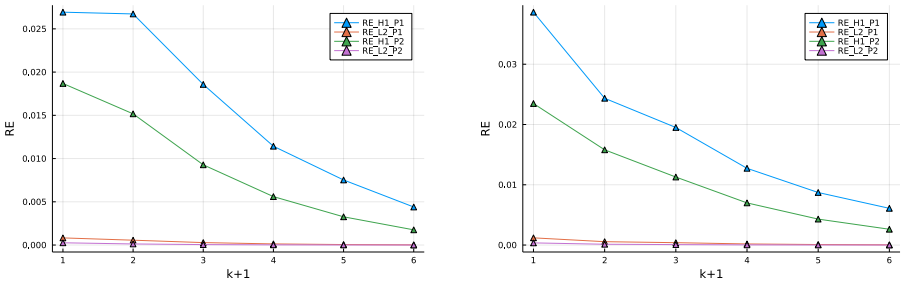


Figure 3.15: Relative error (3.92) measured in the L^2 and H^1 norm of $\mathbf{u}$ on mesh refinements $\{h/2^k, k = 0, .., 6\}$ with $h = 0.125$. All other parameters used correspond to 3.14

The relative error is computed by interpolating $\mathbf{u}_{h_k}$ on the discretization corresponding to h_{k+1}, and the relative error is measured on two curves Λ obtained by the method of CCO with 36 terminal vessels.

With these numerical results, we conclude confidence that the mathematical model exhibits the physical features we expected and that the corresponding numerical procedure is consistent and converges for higher refinement. We have thus a way of describing the interaction of fluid in the vasculature and the surrounding tissue by the mathematical description of diffusion and exchange. By the method of CCO we can obtain the spatial location of the vascular trees which is needed as the input for the machine learning model and by the boundary value problem with fixed boundary conditions described above we obtain the target to the model by using the now investigated finite element solver.

CHAPTER 4

Deep learning with the Fourier neural operator

Many problems in science and engineering involve solving partial differential equations (PDEs) repeatedly for different geometries. The traditional finite element solvers, like the one we introduced in the previous chapter, often rely on a particular kind of discretization to obtain the solution. In situations where we are dealing with complex geometries and particularly in situations where the geometries vary but the PDEs and boundary conditions remain the same this can become costly operations. From our convergence studies in the previous chapter, we clearly see that there is a trade-off between the approximation error and the refinement parameter h. Because of this, there has been an interest in using supervised learning to relax the need for high refinement. A particular neural network architecture achieving tremendous success for computer vision tasks is the convolutional neural network (CNN), by using local convolutions these deep networks have been proven good at detecting local patterns as edges and shapes in images. Since images are a natural way to represent geometries there have been studies on how they can handle computational fluid dynamics (CFD) tasks (see e.g. Guo et al., 2016; Viquerat & Hachem, 2020). However, the primary success of CNNs has been in situations where no mathematical details of the relation between the input image and the output image are known. The deep structures of CNNs then provide enough complexity to learn everything purely by the features of the input data. In learning solutions to PDEs, it has therefore become a question of if you can incorporate a priori knowledge of the problem you are trying to learn into the structure of the neural network. In this way, you could aim to learn an entire family of solutions to PDEs instead of the traditional solvers which obtain the solution for one instance of the PDE. This is known as operator learning and two different approaches in the literature are DeepOnet (Lu et al., 2021) and simply "Operator learning"(Kovachki et al., 2021). In our study we will focus on the latter and specifically Fourier Neural Operators which employ global convolutions through discrete Fourier transforms to learn mappings between function spaces.

In this chapter, we first motivate the use of operator learning to obtain approximations of the solution $(u, \hat{u})$ by considering the energy functional of the previously introduced boundary value problem. We then describe the method proposed in (Li et al., 2020a) to train a parametric map $G_\theta : X \to Y$ working as an approximation to $(u, \hat{u})$ through supervised learning. Lastly, we present our

choice of the input X and the target Y and show it relates to the vasculature Λ and the pressures $(u, \hat{u})$.

Motivation

We define the energy functional

$$E_f(u, \hat{u}) := \frac{1}{2}\int_\Omega (\nabla u)^2 dx + \frac{1}{2}\int_\Gamma (\nabla \hat{u})^2 ds + \frac{\gamma}{2}\int_\Lambda (u - \hat{u})^2 ds \tag{4.1}$$

for $\gamma \geq 0$ and consider the following minimization problem

$$\min_{(u,\hat{u})\in\mathbb{V}} E_f(u, \hat{u}).$$

A minimum $(u, \hat{u})$ is characterized by $E_f(u, \hat{u}) \leq E_f(u + \epsilon v, \hat{u} + \epsilon\hat{v})$ for all $v, \hat{v} \in V$ and $\epsilon > 0$. Defining a new function

$$\Phi(\epsilon) := E_f(u + \epsilon v, \hat{u} + \epsilon\hat{v})$$

we know that Φ has a minimum at $\epsilon = 0$ and $\frac{d\Phi(\epsilon)}{d\epsilon}|_{\epsilon=0} = 0$. Whence,

$$\begin{aligned}\frac{dF(u + \epsilon v, \hat{u} + \epsilon\hat{v})}{d\epsilon} &= \int_\Omega (\nabla u + \epsilon\nabla v) \cdot \nabla v dx \\ &+ \int_\Lambda (\nabla\hat{u} + \epsilon\nabla\hat{v}) \cdot \nabla\hat{v} ds + \gamma\int_\Lambda (u + \epsilon v - (\hat{u} - \hat{v})) \cdot (v - \hat{v}) ds \\ &= 0 \qquad \forall (v, \hat{v}) \in \mathbb{V}.\end{aligned}$$

Setting $\epsilon = 0$ we obtain the familiar bilinear form

$$B[(u, \hat{u}), (v, \hat{v})] = \int_\Omega \nabla u \cdot \nabla v dx + \int_\Lambda \nabla\hat{u} \cdot \nabla\hat{v} ds + \gamma\int_\Lambda (u - \hat{u})(v - \hat{v}) ds. \tag{4.2}$$

We can thus express our boundary value problem as finding the pair $(u, \hat{u})$ which minimizes E_f, and fixing the boundary conditions on Ω and γ and seeking to minimize E_f only dependent on the structure of Λ we have

$$G := \min_{(u,\hat{u})\in\mathbb{V}} E_f(u, \hat{u}; \Lambda), \qquad G(\Lambda) \mapsto (u, \hat{u}). \tag{4.3}$$

Presume now that we have a set of observations $\{X, Y\}_{i=1}^N$ where X and Y correspond to some representation of the vacular geometry and the solution $(u, \hat{u})$ respectively. We know that there exists an operator

$$G : X \to Y \tag{4.4}$$

and aim to learn the operator G by constructing an approximation $G_\theta \approx G$, where $G_\theta : X \times \Theta \to Y$ is a parametric map. More specifically, by defining a cost functional $C : Y \times Y \to \mathbb{R}$ which measures the difference between the output of the operator G and our approximation G_θ, we seek a minimizer $\theta \in \Theta$ by

$$\min_{\theta\in\Theta} C\left[G_\theta(X, \theta), G(X)\right]. \tag{4.5}$$

Provided that G_θ is differentiable w.r.t. θ we can employ optimization algorithms to efficiently find a minimum $\theta \in \Theta$. In the following derivation motivated and inspired by (Li et al., 2020a), (Li et al., 2020b), (Kovachki et al., 2021) we present a specific construction of the parametric map G_θ.

4.1 The method of FNO

Let $\Omega_j = \{\mathbf{x}_j\}_{j=1}^n$ be a discretization of Ω, where $\mathbf{x}_j$ are evenly spaced points. Furthermore, assume we have observations $X \in \mathbb{R}^{n \times d_x}$ and $Y \in \mathbb{R}^{n \times d_y}$, where d_x represents the number of channels in the input X and d_y the number of channels in the output Y.
We then provide the following network architecture, given as an iterative scheme

$$v_0(\mathbf{x}) = P(X(\mathbf{x})) \tag{4.6a}$$
$$v_{t+1}(\mathbf{x}) = \sigma(W v_t(\mathbf{x}) + (\mathcal{K}(Xs;\theta)v_t)(\mathbf{x})) \tag{4.6b}$$
$$u(\mathbf{x}) = Q(v_T(\mathbf{x})). \tag{4.6c}$$

(4.6)a is denoted the lifting layer, (4.6)b is denoted the Fourier layer and (4.6)c is denoted the projection layer. Notice that each iterative step is expressed as function evaluations, and $(\mathbf{x})$ represents that each layer or iterative step is functions taking spatial coordinates as input. $P(\cdot) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_v}$ is denoted the "lifting" operator since we let $\mathbb{R}^{d_v} > \mathbb{R}^{d_x}$, $\sigma(\cdot) : \mathbb{R} \to \mathbb{R}$ is an activation function, $W : \mathbb{R}^{d_v} \to \mathbb{R}^{d_v}$ is a linear transformation and $Q(\cdot) : \mathbb{R}^{d_v} \to \mathbb{R}^{d_y}$ is a projection operator, whose role is to project the output v_T to the desired output shape.

The kernel integral operator is defined by

$$(\mathcal{K}(X;\theta)v_t)(\mathbf{x})) := \int_D \kappa_\theta(\mathbf{x}, \mathbf{s}, X(\mathbf{x}), X(\mathbf{s});\theta)v_t d\mathbf{s}. \tag{4.7}$$

where $\kappa_\theta(\cdot) : \mathbb{R}^{2(2+d_x)} \to \mathbb{R}^{d_v \times d_v}$. To estimate the integral above we impose $\kappa(\mathbf{x}, \mathbf{s}, X(\mathbf{x}), X(\mathbf{s})) = \kappa(\mathbf{x} - \mathbf{s})$ and can then apply the Convolutional Theorem (Renardy & Rogers, 1993, p. 158) to obtain

$$(\mathcal{K}(\theta)v_t)(\mathbf{x})) = \mathcal{F}^{-1}(\mathcal{F}(\kappa_\theta) \cdot \mathcal{F}(v_t))(\mathbf{x}) \tag{4.8}$$

and by direct parameterization of κ_θ in Fourier space, we simplify the above to

$$(\mathcal{K}(\theta)v_t)(\mathbf{x})) = \mathcal{F}^{-1}(R_\theta \cdot (\mathcal{F}v_t))(\mathbf{x}) \tag{4.9}$$

and denote it as the Fourier integral operator. Notice that the operator is now independent of the input X but we see from the architecture (4.6) that both $W \neq 0$ and v_t will "carry" the information of X through the different layers.
Let $f(\mathbf{x}) : \Omega \to \mathbb{R}$ be an integrable function. The Fourier transform of $f(\mathbf{x})$ is defined as

$$\mathcal{F}(f)(\mathbf{k}) = \int_\Omega f(\mathbf{x}) e^{-2\pi i \langle \mathbf{x}, \mathbf{k} \rangle} d\mathbf{x}. \tag{4.10}$$

where $i = \sqrt{-1}$ is the imaginary unit and $\langle \cdot, \cdot \rangle$ denotes the inner product between the two d_x-dimensional vectors $\mathbf{x}$ and $\mathbf{k}$. Since our domain Ω is discretized with points $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ (4.10) is approximated by the discrete Fourier transform expressed as

$$\mathcal{F}f_l(\mathbf{k}) = \sum_{j=1}^{n} f_l(\mathbf{x}_j) e^{-2\pi i \langle \frac{\mathbf{x}_j}{n}, \mathbf{k} \rangle} = \sum_{j_x=1}^{n_x} \sum_{j_y=1}^{n_y} f_l(x_{j_x}, y_{j_y}) e^{-2\pi i \left[\frac{x_{j_x}}{n_x} k_x + \frac{y_{j_y}}{n_y} k_y \right]} \tag{4.11}$$

where $l = \{1, .., d_v\}$. Calculating $\mathcal{F}v_t$ as above using the Fast Fourier Transform (Cooley & Tukey, 1965) for all modes $\{\mathbf{k}_1, \mathbf{k}_2, ..., \mathbf{k}_n\}$ and all channels or dimensions l we obtain $\mathcal{F}v_t \in \mathbb{C}^{d_v \times n}$ [1].

Following (4.6) the next step is to multiply the Fourier transform $\mathcal{F}v_t$ to a weights tensor R_θ. The number of learnable parameters in R_θ is given by $(d_v \cdot d_v \cdot n)$ complex numbers and we truncate the higher modes by letting $\mathcal{F}v_t \in \mathbb{C}^{d_v \times \mathbf{k}_{max}}$ where $\mathbf{k}_{max}$ is the maximum number of modes to keep, we denote these modes the cut-off frequencies. R_θ is then reduced to $R_\theta \in \mathbb{C}^{d_v \times d_v \times k_{max}}$ and the multiplication is defined by

$$(R_\theta \cdot \mathcal{F}v_t)_{k,l} = \sum_{j=1}^{d_v} R_{l,j,k}(\mathcal{F}v_t)_{k,j}, \qquad k = 1, .., k_{max}, \;\; j = 1, .., d_v. \quad (4.12)$$

Notice that we restrict ourselves to the case where the maximum number of modes to keep is the same for each dimension in Ω, i.e. $k_{max,x} = k_{max,y}$ for $\mathbf{k}_{max} = (k_{max,x}, k_{max,y})$.

Similarly as (4.11) we have the discrete inverse Fourier transform defined as

$$\mathcal{F}^{-1} f_l(x) = \frac{1}{n} \sum_{j=1}^{n} f_l(x_j) e^{2\pi i \langle \frac{k_j}{n}, x \rangle}. \quad (4.13)$$

After obtaining $(R_\theta \cdot \mathcal{F}v_t) \in \mathbb{C}^{d_v \times k_{max}}$ we compute the inverse Fourier transform (4.13) by zero-padding $(R_\theta \cdot \mathcal{F}v_t)$ before the transformation to obtain the desired output shape $\mathcal{F}^{-1}(R_\theta \cdot \mathcal{F}v_t) \in \mathbb{R}^{n \times d_v}$.

4.2 Model architecture and parameter choices

We summarize the neural network architecture and parameters for the method introduced in the previous section. The restriction on the lifting and projection

Identity	Restriction	Map
$P(\cdot)$	$P(X(x)) = (P(X))(x)$	$\mathbb{R}^{d_x} \to \mathbb{R}^{d_v}$
$Q(\cdot)$	$Q(v_T(x)) = (Q(v_T))(x)$	$\mathbb{R}^{d_v} \to \mathbb{R}^{d_y}$
W	linear transformation	$\mathbb{R}^{d_v} \to \mathbb{R}^{d_v}$
$\sigma(\cdot)$	nonlinear transformation	$\mathbb{R} \to \mathbb{R}$
σ_{MLP}	nonlinear transformation	$\mathbb{R} \to \mathbb{R}$

Table 4.1: Transformation specifications for network architecture.

operators $P(\cdot), Q(\cdot)$ is that they should be pointwise operations for any $\mathbf{x} \in \Omega_j$. We will consider $P(\cdot), Q(\cdot)$ defined as an multilayer perceptron with activation $\sigma_{MLP}(\cdot)$ and $h_{MLP} \leq 2$ number of hidden layers with size d_{MLP} and restriction $d_{MLP} \geq d_v$. To illustrate how the restriction in Table 4.1 is satisfied we let $P(\cdot)$ be defined with $h_{MLP} = 1$ and can then express the operator as

$$P(X(\mathbf{x})) = (\sigma_{MLP}(X(x)W_1^T + b_1)W_2^T + b_2 \quad (4.14)$$

[1] Since the Fourier transform of real input is Hermitian-symmetric, we only collect the modes leading up to the Nyquist frequency, hence n is in fact $N_x \times (N_y//2 + 1)$.

where W_i represents the weights and b_i the bias. Since and $W_1 : \mathbb{R}^{d_x} \to \mathbb{R}^{d_{MLP}}, b_1 \in \mathbb{R}^{d_{MLP}}$, $W_2 : \mathbb{R}^{d_{MLP}} \to \mathbb{R}^{d_v}$ and $b_2 \in \mathbb{R}^{d_v}$ and since $\sigma_{MLP}(\cdot)$ action is defined pointwise we see that $P(X)$ is independent of the discretization n and fulfill the restriction in table 4.1. The same reasoning generalizes to $h_{MLP} > 1$ and the same argument holds for $Q(\cdot)$. Section 4.1 together with

Data features	d_x		number of channels in input X
	d_y		number of channels in output Y
	n		number of points in discretion Ω_j
	N		number of data samples
Model features	d_v		number of channels in $v_i(x)$
	d_{MLP}		width of hidden layer in MLP
	k_{max}		max number of frequency modes
	T		number of Fourier layers v_i

Table 4.2: Parameter specifications for the model and the data.

Table 4.1 and Table 4.2 provides the necessary details to construct the parametric map G_θ. One key feature of G_θ is that it is discretization independent, i.e. we can train the model on one resolution and predict on another. We do however restrict ourselves to square discretization $N_x = N_y$.

4.3 Input representation

To obtain the input X we use the geometries obtained by the method of CCO. By the model reduction in section 3.2 we have simplified the geometries to 1D and hence model the "vessels" purely by points and connections. To obtain these geometries we let Ω, the representation of the tissue be the unit square $(1,0)^2$ and initiate the CCO algorithm with a root segment $(\_, x_0, x_1)$. We draw x_0 from a random uniform distribution along the left boundary edge of $\partial\Omega$ i.e. the line segment between $(0,0)$ and $(0,1)$. x_1 is then drawn inside a a padded domain Ω_p until it fulfills the minimum distance criteria (2.7). The growth of the arterial tree is always restricted to the domain Ω_p with vertices $(0+p, 0+p), (1-p, 0), (1-p, 1-p)$ $(0+p, 1-p)$, where p represents a padding factor with the purpose of only allowing x_0 to be on the boundary $\partial\Omega$. The arterial tree is generated until the number of terminal segments $N_{term} = N^t_{final}$. By observation, considering the features; complexity, randomness, verisimilitude and computing time the remainder parameter specifications as presented in Table 4.3 are used to obtain the data set $\{\mathcal{T}_k\}_{k=1}^M$.

Note that the root radius r_0 is a dimensionless quantity decided as a fraction of Ω and is required to generate the trees $\{\mathcal{T}_k\}_{k=1}^M$ but obsolete in the 1D geometry representation of the trees $\{\Lambda_k\}_{k=1}^M$. Indeed, we have $\{\Lambda_k\}_{k=1}^M$ represented by the points corresponding to the distal and proximal points of the vessels and the connections $(\mathrm{id}_{x_p}, \mathrm{id}_{x_d})_i$ representing the vessels by the indices of the proximal and distal point to vessel v_i. The parameters in Table 4.3 will be fixed for every experiment but we will vary the number of terminal segments N^t_{final}.

We aim to learn the operator G in (4.3) by a neural network approximation $G_\theta : X \to Y$ where X denotes the input representation of the 1D geometry Λ and Y is the output representation of the solution $(u_h, \hat{u}_h)$. In the simplest

Geometric	γ	3.0	bifurcation exponent
	δ	0.7	symmetry ratio parameter
	p	0.01	padding factor
Physical	r_0	0.01	root radius
Tuning	ν	1.0	tuning parameter
	f_r	0.9	correction step factor
Optimization	N_{fail}	10	correction number
	N_{con}	3	number of vessels evaluated in connection search
	Δv	6	discretion parameter for bifurcation point search

Table 4.3: Parameter specifications for the method of CCO.

two-dimensional form, Λ could then be represented as a binary $N_x \times N_y$ matrix taking ones on the curve Λ and zeros elsewhere, or the opposite. However, following (Guo et al., 2016) we see that using a distance function to represent the location of Λ in the square domain Ω could obtain better results when training a neural network. It is additionally somewhat intuitive that more information in the input helps the training of the model. We therefore let $X = D_\Lambda$ be a distance representation of the 1D geometry Λ by

$$D_\Lambda : \mathbf{p} \in \Omega \rightarrow \min_{\mathbf{q} \in \Lambda} |\mathbf{p} - \mathbf{q}| \tag{4.15}$$

where $\mathbf{p}$ and $\mathbf{q}$ are points in the plane Ω. To obtain the minimum distance from a point in the plane Ω to a point on the geometry Λ we recall that Λ is made up of sets of points and connecting lines. The equation of a line between two points $\mathbf{p}_1, \mathbf{p}_2$ is $y = \mathbf{p}_1 + a(\mathbf{p}_2 - \mathbf{p}_1)$ and the shortest distance from the point $\mathbf{p}_3$ to the line y is given by the criteria $(\mathbf{p}_3 - y) \cdot (\mathbf{p}_2 - \mathbf{p}_1) = 0$ i.e. when the line connecting $\mathbf{p}_3$ is perpendicular on y. Inserting the equation of the line and solving for a we obtain

$$a = \frac{(x_3 - x_1)(x_2 - x_1) + (y_3 - y_1)(y_2 - y_1)}{\|\mathbf{p}_2 - \mathbf{p}_1\|^2} \tag{4.16}$$

which gives the point of intersection $\mathbf{p}_{int} = \mathbf{p}_1 + a(\mathbf{p}_2 - \mathbf{p}_1)$ and hence the distance from a point $\mathbf{p}_3$ to a line y passing through two points $\mathbf{p}_1$ and $\mathbf{p}_2$ is given by

$$\text{dist}(y, \mathbf{p}_3) = \|\mathbf{p}_3 - \mathbf{p}_{int}\|. \tag{4.17}$$

By setting

$$a = \begin{cases} 1 & a > 1 \\ 0 & a < 0 \\ 4.16 & \text{else} \end{cases}$$

we obtain the shortest distance from a point $\mathbf{p}_3$ to a line *segment* between the points $\mathbf{p}_1$ and $\mathbf{p}_2$.
Calculating the above for all line segments making up Λ and choosing the minimum distance we obtain the distance from a point $\mathbf{p}$ to Λ. D_Λ is then obtained by doing this calculation for evenly spaced points in Ω where the spacing of the points decides the resolution $N_x \times N_y$.

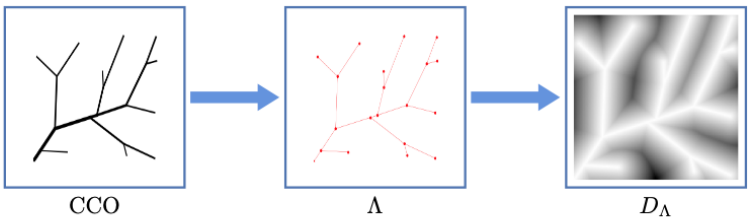


Figure 4.1: Flow chart for obtaining distance representation of 1D geometry Λ.

Figure 4.1 illustrate the steps involved to obtain the input X to the network G_θ and we summarize them as follows.

1. Obtain vascular tree with fixed parameters provided in Table 4.3 and choice of N^t_{final}.

2. Extract points and connecting lines to obtain Λ.

3. Compute distance representation of Λ by the method described previously and with specified resolution $N_x \times N_y$.

4.4 Target representation

To obtain the target representation Y we solve the previously introduced boundary value problem as in section 3.6. With prescribed Dirichlet condition $D = \frac{1}{1+d_i}$ and homogeneous Neumann conditions on Σ. The appropriate trial and test spaces are then

$$\mathcal{U} = H^1(\Omega) \times H^1_D(\Lambda)$$

$$\mathcal{V} = H^1_0(\Omega) \times H^1_0(\Lambda)$$

and we recall the boundary value problem of finding $(u, \hat{u}) \in \mathcal{U}$ s.t.

$$\int_\Omega \nabla u \cdot \nabla v + \int_\Gamma \nabla \hat{u} \cdot \nabla \hat{v} + \int_\Lambda \gamma(u - \hat{u})(v - \hat{v}) = 0 \quad \forall (v, \hat{v}) \in \mathcal{V}. \tag{4.18}$$

Following the derivations made in section 3.4 we then seek the discrete solutions $(u_h, \hat{u}_h)$ defined on the partitioning $(\mathcal{P}_h, \mathcal{I}_h)$.
To represent the solution $\hat{u}_h$ we let $E\hat{u}$ be defined as the harmonic extension of the solution $\hat{u}_h$ by

$$-\Delta E\hat{u} = 0 \qquad \text{on } \Omega \tag{4.19}$$

$$E\hat{u} = \hat{u}_h \qquad \text{on } \Lambda \tag{4.20}$$

$$-\nabla E\hat{u} \cdot n = 0 \qquad \text{on } \partial\Omega. \tag{4.21}$$

$E\hat{u}$ will then be a 2D solution defined on Ω and identical to $\hat{u}_h$ on the partitioning $\mathcal{I}_h$ of Λ. To obtain the solution $E\hat{u}_h$ we follow the standard approach of deriving

the variational formulation of the above boundary value problem, we multiply with test functions $v^{he} \in V^{he}$ and integrate over the domain Ω. By the last term above (zero Neumann), the exterior boundary term vanishes and we get: Find $E\hat{u} \in U^{he}$ s.t.

$$\int_{\Omega} \nabla E\hat{u} \cdot \nabla v^{he} = 0 \qquad \forall v^{he} \in V^{he}. \tag{4.22}$$

The enforcement of the Dirichlet boundary condition on the interior boundary Λ is specified by letting $U^{he} = \{H^1(\Omega) : E\hat{u} = \hat{u} \text{ on } \Lambda\}$ and $V^{he} = \{H^1(\Omega) : E\hat{u} = 0 \text{ on } \Lambda\}$ and after obtaining the solutions $u_h, E\hat{u}_h$ both on the triangulation $\mathcal{P}_h$ we interpolate to obtain the desired square resolution $N_x \times N_y$.

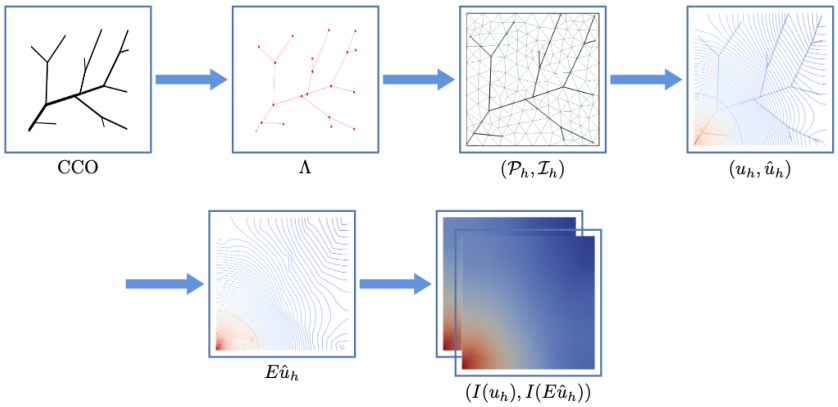


Figure 4.2: Flow chart for obtaining the target representation Y.

Figure 4.2 illustrates the procedure of obtaining the target representation to the network G_θ, we summarize them as follows, since the first two steps are shared with obtaining the input we only state the remainder.

3. Partition geometry (Ω, Λ).

4. Implement and solve linear system (3.62) corresponding to (4.18).

5. Use solution $\hat{u}_h$ to obtain the harmonic extension $E\hat{u}_h$ by solving (4.22).

6. Interpolate u_h and $E\hat{u}_h$ to desired resolution $N_x \times N_y$.

Since the two first steps corresponding to obtaining the geometry Λ are shared between the input and target representation, we combine the two steps to generate the data set $\{X, Y\}_{i=1}^N$. This allows for efficient implementation and also ensures that the input and target samples agree. A specific attribute of the Fourier neural operator is that it takes function evaluations as input, hence the input X corresponding to D_Λ should be presented to the network as $X(\mathbf{x})$ where $\mathbf{x}$ is the points in the discretization Ω_j. Since $\mathbf{x}$ is equal for all samples X_i we simply append $\mathbf{x}$ to each sample X_i before we present it to the network. Since our configuration of the CCO algorithm in 2.1 and specifically the choice of root proximal point x_0 provides that the different 1D structures do *not*

exhibit rotational symmetry we can increase our data with consecutive 90-degree rotations, rotating the input X and the corresponding output Y. Data augmentation by rotations then quickly quadruples the number of samples in the data set.

4.5 Optimization

Having a model for the parametric map G_θ, and a way to obtain the data samples $\{X, Y\}_{i=1}^N$ we now want to construct a routine for obtaining the set of parameters $\theta \in \Theta$ which correspond to a minimum of the cost (4.5). Defining a specific way to evaluate the cost which is differentiable with respect to the parameters, the gradient will contain information on the direction and rate of fastest increase. We can then update the parameters in the opposite direction of the gradient, and the following evaluation of the cost should produce a value smaller than the previous evaluation. On a perfect convex cost function, doing this procedure iteratively we will obtain the minimum θ. In the case of machine learning and neural networks, the cost is not simply a convex curve, but a multi-dimensional surface with peaks and valleys, however, the same principle applies, we always want to update the parameters in such a way that it reduces the cost. Doing this in an iterative procedure denotes optimization, and the specific choice of optimization leads to an optimization algorithm. There are multiple different ways to evaluate the cost and also optimization algorithms to choose from. Following a similar model structure as (Li et al., 2020a) it is natural to choose a similar optimization algorithm and cost evaluation. This also has the benefit of easily being able to compare our results to that of (Li et al., 2020a) and (Li et al., 2020b).

Loss function

Since the difference of the output G_θ and the target Y will be evaluated for subsets of the data $\{X, Y\}_{i=1}^N$ we use the terminology of loss function instead of cost function and define it as the relative L^2 loss by

$$L(G_\theta(X), Y) = \frac{1}{|\Omega_j|} \frac{\|G_\theta(X) - Y\|_{L^2(\Omega_j)}}{\|Y\|_{L^2(\Omega_j)}} \tag{4.23}$$

The norm is calculated using the pointwise squared absolute error

$$\|Y\|_{L^2(\Omega_j)} = \sqrt{\sum_{j=1}^{n} |Y_j|^2}, \tag{4.24}$$

and the average over points in the discretization allows us to compare the loss for different resolutions.

Optimization Algorithm

Since its introduction in (Kingma & Ba, 2017) the Adam optimization algorithm has been a popular choice for machine learning tasks and is indeed the optimization algorithm chosen in (Kovachki et al., 2021).

Algorithm 2 Adam

Data: $\beta_1, \beta_2 \in [0, 1), \lambda, \theta_0, L, \epsilon$

$t = 0$
$m_0 = 0$
$v_0 = 0$
while $t \leq T$ **do**
  $t = t + 1$
  $g_t = \nabla_\theta L(G_{\theta_{t-1}}(X), Y) + w\theta_{t-1}$
  $m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$
  $v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$
  $\hat{m}_t = m_t / (1 - \beta_1^t)$
  $\hat{v}_t = v_t / (1 - \beta_2^t)$
  $\theta_t = \theta_{t-1} - \lambda \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$
end while

Algorithm 2 describes the Adam optimization algorithm and how the gradient of the model is used to update the parameters θ. β_1, β_2 are hyperparameters used to compute running averages of the gradient g_t and its square g_t^2, λ is the stepsize/learning rate, θ_0 is the initial state of the parameters, w is a L^2 regularization term as described in (Loshchilov & Hutter, 2017) and ϵ is a term added to prevent division by a number ≈ 0 and hence improve numerical stability. We do not present the way to obtain the value $\nabla_\theta L$ as this is conveniently done through the Pytorch software (Paszke et al., 2019) using automatic differentiation.

Weights and bias initialization

Lastly, we need to define our initial guess of the parameters θ, we denote this as the choice of *initialization* for the weights and bias. Choosing an appropriate initialization can help the model G_θ to more quickly learn the generalized features and to prevent exploding/vanishing gradients. For $P(\cdot)$, $Q(\cdot)$ and W we will use LeCun initialization (LeCun et al., 2002) defined as samples drawn from the uniform distribution

$$U\left(-\sqrt{\frac{1}{n_{in}}}, \sqrt{\frac{1}{n_{in}}}\right). \tag{4.25}$$

The values in the parenthesis denote the range in which the numbers are drawn and n_{in} is the number of input features, i.e. $n_{in} = d_x$ for $P(\cdot)$ and $n_{in} = d_v$ for $Q(\cdot)$ and W. This is the default PyTorch initialization for linear transformations e.g. 2D convolutional layers with kernel size (1×1).
The complex values of R_θ are drawn from

$$U\left(0, \frac{1}{n_{in} \cdot n_{out}}(1 + i)\right) \tag{4.26}$$

where $n_{in} \cdot n_{out}$ represents the multiplication of the number of input and output features of the Fourier layer. This choice of this initialization is simply based on the implementation from (Li et al., 2020a).

4.6 Training the model

Having a data set $\{X, Y\}_{i=1}^{N}$, a specific initialization of the parameters θ, a way to measure the difference of the prediction $G_\theta(X)$ and the target Y and a way to update the parameters θ we combine all and denote it "training the model". We split the data set $\{X, Y\}_{i=1}^{N}$ into two respective groups, the training set and the test set. The training set is used to update the parameters θ and the test set is used for an unbiased evaluation of model prediction during training. This prevents the parameters θ to learn the specific features in the training set (overfitting) but rather the generalized features of the data. To save memory requirements we follow the standard procedure of splitting the data into batches to evaluate the model and compute the gradient. The learnable parameters θ are then updated via the optimization algorithm Algorithm 2 for every batch in the training set and this procedure is repeated for every epoch in N_{epochs}.

Algorithm 3 Training

Data: $N_{epoch}, L, Adam(\cdot, \beta_1, \beta_2, \lambda, L, \epsilon), \{X, Y\}_{test}, \{X, Y\}_{train}$

$i = 0$
while $i \leq N_{epoch}$ **do**
  $i = i + 1$
  $j = 0$
  for $(x, y) \in \{X, Y\}_{train}$ **do** ▷ Randomly choose batch from training data
    $y_{pred} = G_{\theta_j^i}(x)$ ▷ Make prediction using θ_i
    $l_{train} = L(y_{pred}, y)$ ▷ Compute loss by (4.23)
    $L_{batch,train} \leftarrow l$ ▷ Store loss
    $\theta_{j+1}^i = Adam(\theta_j^i, \beta_1, \beta_2, \lambda, L, \epsilon)$ ▷ Update parameters as in Algorithm 2
    $j = j + 1$
  end for
  $L_{train}^i = \frac{1}{N_{batch}} \sum_{k=1}^{N_{batch}} L_{k,train}$

  for $(x, y) \in \{X, Y\}_{test}$ **do** ▷ Evaluate parameter configuration θ_i on test data
    $y_{pred} = G_{\theta^i(x)}$
    $l_{test} = L(y_{pred}, y)$
    $L_{batch,test} \leftarrow l$
  end for
  $L_{test}^i = \frac{1}{N_{batch}} \sum_{k=1}^{N_{batch}} L_{k,test}$

  if $l_{test}^i < L_{test}^{i-1}$ **then**
    $\theta_{best} = \theta^i$ ▷ Store parameter configuration
  end if
end while

Algorithm 3 shows the training procedure of obtaining the best model G_θ. We randomly shuffle the batches in the training and test set before we draw the batches needed for updating the parameters θ and evaluating the model.

CHAPTER 5

Numerical experiments

Through a series of experiments, we want to investigate the learnability of the proposed parametric map $G_\theta : X \to Y$. We will restrict ourselves to a maximum number of 10000 data samples $\{X, Y\}_{i=1}^N$, but will produce some additional data to evaluate different features of the model G_θ. Our aim is to provide a *proof of concept* study of the model G_θ, introduced in the previous chapter ability to learn the generalized features in the data $\{X, Y\}_{i=1}^N$. These features correspond to the operator described as the minimization of the energy functional (4.1) of the boundary value problem (4.18). The features change when varying the geometry Λ, but the generalized features do not.

The most similar example of our problem description is the Darcy Flow example provided in (Li et al., 2020a), where they consider the steady state Darcy's equation in two dimensions for different values on the diffusion coefficient. Since this example provides a reference of the structure on G_θ and corresponding results, we first simplify our problem setup such that it resembles the Darcy flow example with only one image target. Finding a specific configuration on this simplified model we add an additional channel in the target Y and evaluate different model architectures allowing for the two-image prediction. Using the best-obtained model we complete the study by looking at specific features provided by the alternative map $G_\theta : X \to Y$ and the time-sparing capability compared to the traditional finite element solver. Lastly, we make our concluding remarks and propose how the work presented in the thesis can be extended.

5.1 Experiment 1: One-image target

From the physical interpretation provided in section 3.6 a geometry Λ with a high number of terminal segments will result in a solution $u \in \Omega$ with a homogeneous distribution of the pressure and a geometry Λ with a small number of terminal segments yields a less homogeneous distribution. The impact of the number of terminal segments on the pressure distribution is therefore much more visible for a few numbers of terminal segments and it is reasonable to presume that the map between Λ and u is easier to learn. Motivated by this we first aim to build the approximation G_θ for $2 - 3$ terminal segments in the input data and the corresponding output data.

The three first plots in Figure 5.1 show the input to the network. The embeddings illustrate how the network G_θ consists of layers that are considered

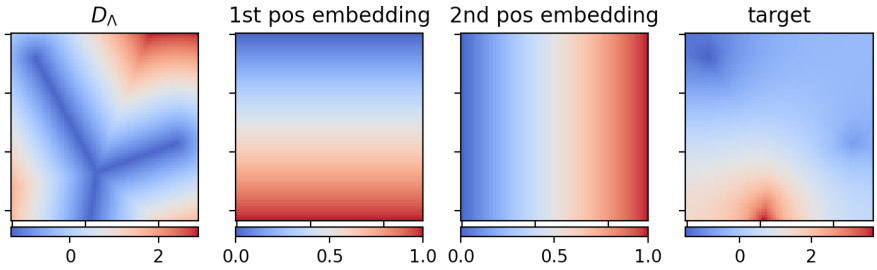


Figure 5.1: Input and target representation, one-image target.

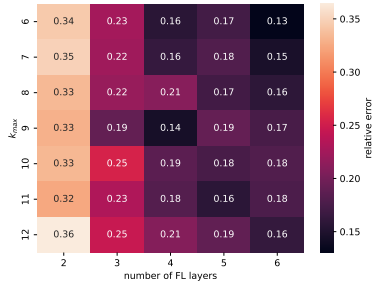
as function taking the points of Ω_j as input. Motivated by our observation in Appendix A, where we see the convergence of u_h to the harmonic extension $E\hat{u}_h$ for strong coupling factor γ, we choose the target as $E\hat{u}_h$ where the location of the terminal segments is most profound.
From chapter chapter 4 it is clear that the network is resolution independent i.e. the discretization of Ω_j does not affect the number of learnable parameters. Since it does not require high resolution to resolve the characteristics of the input D_Λ and the target $E\hat{u}_h$ train the network at 64x64 resolution.
Inspired by the Darcy flow example mentioned previously we first create 1000 training samples and 250 test samples. The data is normalized by extracting the mean and dividing it by the standard deviation. The normalization of the training and test data is done separately. We want to see how the number of parameters θ effects the learnability of the features provided in the test and training set.

We include batch normalization and use only ReLu activation functions after each Fourier layer. In all fully-connected layers we use GeLu activation functions, e.g. in P, Q. Unless otherwise specified we use the Adam optimizer Algorithm 2 with a learning rate 0.001 that is halved for every 100 epoch and we train for a total of 500 epochs. All losses are relative errors by (4.23) and taken as the mean over all samples.

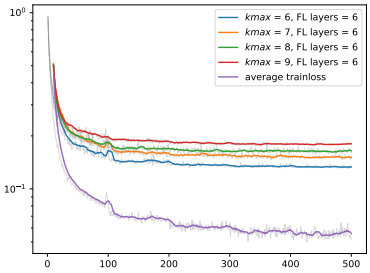
Fourier layers

Since the total number of learnable parameters θ is most affected by the construction of the Fourier layer we first evaluate how increasing the number of cut-off frequencies and the number of Fourier layers decreases the error. We choose $P(\cdot)$ as pure lifting, i.e. 2D convolution with kernel size 1, lifting the dimension of the input to $d_v = 32$. The projection is pure projection and the reverse of the lifting operator, i.e. $Q(\cdot) : \mathbb{R}^{d_v} \to \mathbb{R}^{d_y}$, where currently $d_y = 1$.

Figure 5.2(a) shows the lowest obtained error on the test data when trained on 1000 samples for 500 epochs. The total number of learnable parameters ranges from approximately $4 \cdot 10^2$ to $4 \cdot 10^5$. By increasing the number of learnable parameters in a machine-learning model the model possesses more expressiveness in the form of having more ability to reproduce the features in the target. This implies that the training loss decreases as the total number of learnable parameters increases. However, in learning the generalized features contained in the training set i.e. reducing the test loss, we see from Figure 5.2 (a) that the lowest loss is obtained from a higher number of Fourier layers, but



(a) Lowest obtained test loss for different numbers of cut-off frequencies and numbers of Fourier layers.



(b) test losses corresponding to the top right corner of (a) and the average of the corresponding train losses.

Figure 5.2: Increasing the number of learnable parameters, by increasing the number of cut-off frequencies and the number of Fourier layers.

not necessarily a high number of cut-off frequency modes. Figure 5.2 (b) shows the training loss and test loss during training and plotted as moving averages. We clearly see the case of overfitting on the features contained in the training data. We suspect that the reason for this is that the data does not contain enough samples for the model to learn the generalized features of the problem. We also note that the loss is affected by the initial configuration of the weights and biases in the network as well as the initial samples drawn from the data set. Figure 5.2 (a) should therefore not be used to choose the exact value of k_{max} and the number of Fourier layers, but rather to analyse the impact of the two different ways of increasing the number of learnable parameters.
In this first configuration of G_θ we only considered the number of Fourier layers and the number of cut-off frequencies. Choosing $k_{max} = 6$ and six Fourier layers we want to see if changing the lifting and projection layers could further reduce the test error.

Lifting and projection layer

Examples in the (Li et al., 2020a) show that $P(\cdot)$ are usually chosen as a pure lifting operator i.e. a fully connected layer lifting the dimensionality of the input to a specified number d_v. However, comparing the input to (Li et al., 2020a) Darcy flow problem, we see that they use a binary representation of the input. Because of this difference in input representation, we want to study the impact of different configurations on $P(\cdot)$. We let $P(\cdot)$ be either a pure lifting operator or a multi-layer perceptron with one or two hidden layers with the width given as a multiple of d_v. Similarly, we study the effect of changing the projection operator $Q(\cdot)$. From Figure 5.2 (b) we see that the change in error is small after 200 epochs and therefore we only evaluate the impact on the error for different lifting and projection layers for the first 200 epochs.

Table 5.1 shows the lowest obtained test loss for different configurations on the lifting and projection operators. These operators are always multi-layer perceptrons and the increase in complexity is by adding one or two hidden layers with specified size. The increase in the number of parameters θ for different

(a) Lifting operator

h/w	d_v	$2 \cdot d_v$	$4 \cdot d_v$
0	0.16		
1	0.15	0.16	0.14
2	0.14	0.14	0.16

(b) Projection operator

h/w	d_v	$2 \cdot d_v$	$4 \cdot d_v$
0	0.16		
1	0.15	0.15	0.15
2	0.15	0.16	0.16

Table 5.1: Lowest test loss for different lifting and projection operators, $d_v = 32, k_{max} = 6$ and six Fourier layers.

configurations on P and Q is at most $2 \cdot 10^4$.

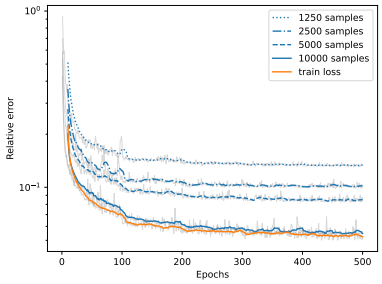
d_v	10	20	30	40	50
min. train loss	0.14	0.09	0.06	0.06	0.05
min. test loss	0.25	0.16	0.16	0.15	0.15

Table 5.2: Minimum loss for increasing number of d_v, $k_{max} = 6$, six Fourier layers, pure lifting and pure projection.

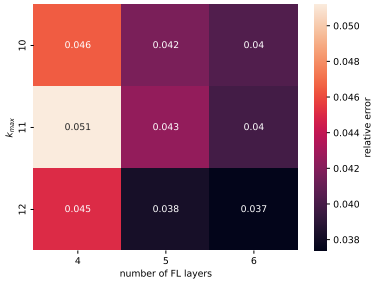
Table 5.2 shows the effect on the lowest obtained training and test loss for increasing the dimensionality of the input, the number of learnable parameters ranges from 10^3 to $2 \cdot 10^5$.

Figure 5.2 (a), Table 5.1 and Table 5.2 shows that we are not able to decrease the test loss further than approximately 15% relative error when changing the architecture of G_θ we therefore conclude that the model needs more data to learn the generalized features, and we increase the data set.

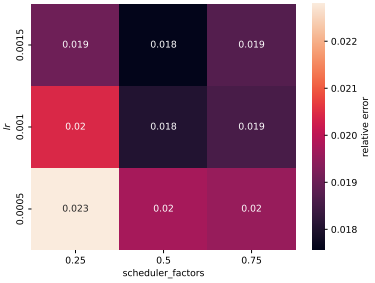
Figure 5.3 (a) shows how increasing the data set further reduces the test loss all the way to the situation where the test and training loss is approximately equal. The plot is obtained using $k_{max} = 6$, six Fourier layers and pure lifting and projection with $d_v = 32$. Ideally, we want the model to have enough learnable parameters available to overfit slightly on the training data, this way we know that the model has enough learnable parameters available to learn the generalized features present in the training data. Figure 5.3 (b) shows the impact on the test error by increasing the number of learnable parameters by increasing k_{max} and the number of Fourier layers. Using $k_{max} = 10$ and six Fourier layers, Figure 5.3 (c) investigates the impact of the learning rate and the learning rate scheduler, we take advantage of the higher number of training samples to approximate the effect on the optimization parameters. Motivated by 5.3 (b) and (c) we retain the configuration on the optimization algorithm i.e. learning rate 0.001 and halving this every 100 epochs but we include a weight decay $w = 10^{-4}$ in Algorithm 2. The learnable parameters are increased by choosing $k_{max} = 10$ with six Fourier layers, $Q(\cdot)$ is chosen as a multilayer perceptron with 1 hidden layer and width $2 \cdot d_v$. $P(\cdot)$ remains a pure lifting operator and $d_v = 32$. Figure 5.3 (d) shows the results obtained with this configuration on the model and the optimization algorithm, the relative error on the training data surpasses 2%, while the relative error on the test data converges to 4%.



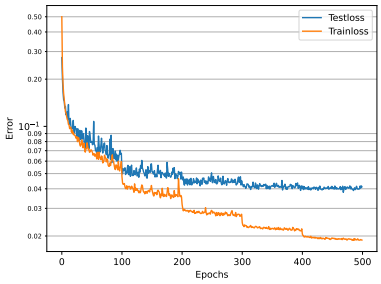
(a) Increasing the data.



(b) Model parameter search, errors on the test set.



(c) Optimization parameter search, errors on the training set.



(d) Best obtained learning curve.

Figure 5.3: Learning the generalized features.

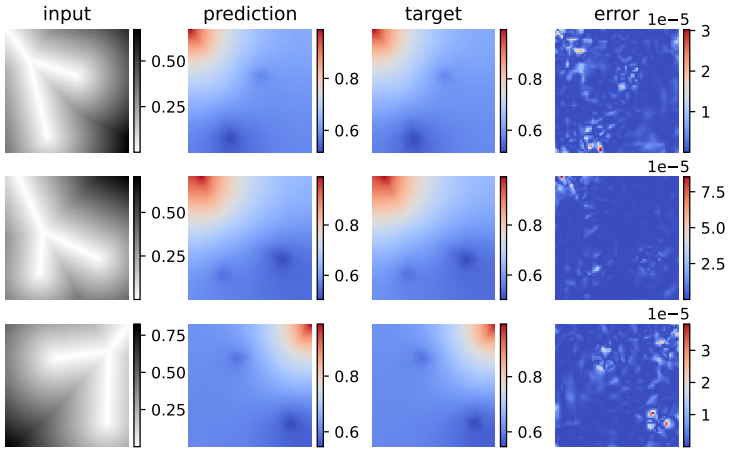


Figure 5.4: Top three predictions. Error is pointwise squared absolute.

Figure 5.4 and Figure 5.5 show the three best and three worst predictions on the 2000 test data samples. We see that the best prediction is on "Y-shaped"

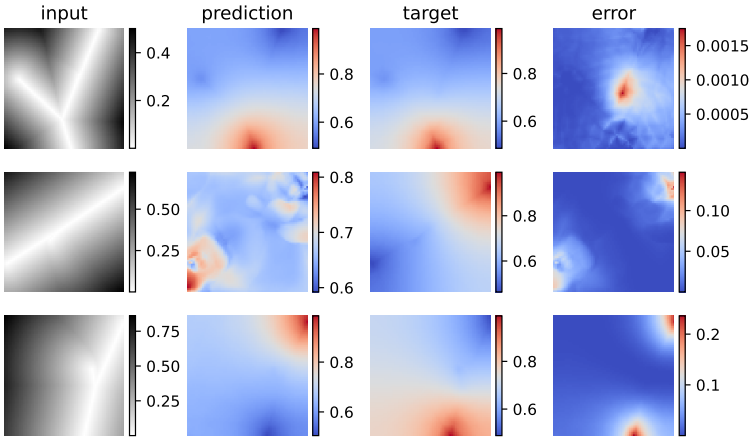


Figure 5.5: Worst three predictions. Error is pointwise squared absolute.

curves Λ, with a high bifurcation angle. The two worst predictions are different in the way that it is harder to locate the specific bifurcation point and the following branching of the curve. Since the bifurcation indicates which part of the graph is the root and which part is the terminal vessel, we see that the model either misses or can't decide which endpoint of the curve that should take the initial value of 1.0. We also note that the method of CCO prefers "Y-shaped" bifurcations and hence shapes of Λ with less prominent bifurcation are underrepresented in the data set.

5.2 Experiment 2: Two-image target

The boundary value problem (4.18) has a solution pair $(u, \hat{u})$ and in experiment 1. we restricted ourselves to the case where we only learn the 2D extension of $\hat{u}$ i.e. the harmonic extension $E\hat{u}$ (4.21). In this experiment, we want to change the model architecture of G_θ such that it predicts the solution u_h for fixed $\gamma = 10$ and additionally $E\hat{u}$. The model is then $G_\theta : X \to Y$, where X remains the distance representation of Λ with its embeddings and Y is a two-image target where the first image is the interpolation of the solution u_h and the second is the interpolation of $E\hat{u}_h$. Because of this extra channel the loss in Algorithm 3 is calculated for both images and the mean of these is used in the optimization Algorithm 2.
There are different ways of restructuring the model G_θ to allow for a two-image output and since experiment 1. already shows the ability to learn the features of one image, a natural restructuring is to simply feed the input X to two respective networks G^1_θ and G^2_θ, we then know the ability to learn the second image $I(E\hat{u}_h)$ and the remaining question is the learnability of $I(u_h)$.

Another approach is to provide some connectivity between the layers, since the solution does indeed come from the same operator G it is natural to presume that parameters θ could be shared between G^1_θ and G^2_θ.

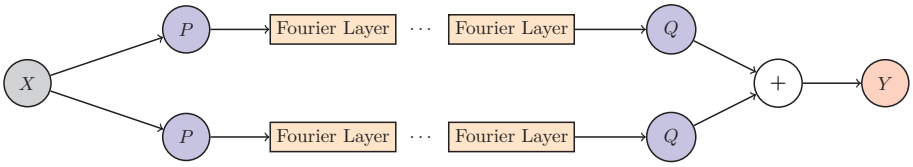


Figure 5.6: Model configuration 1.

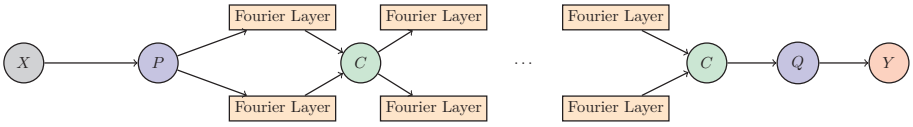
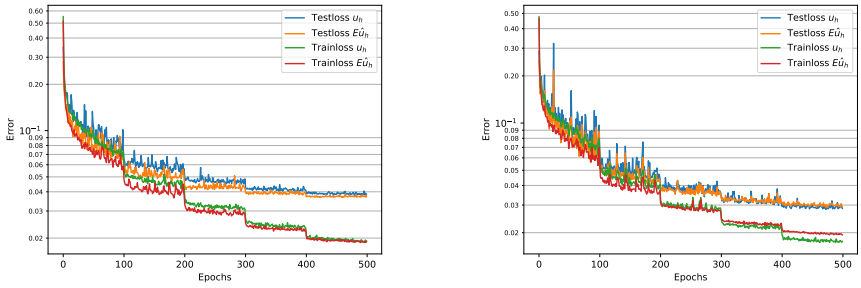


Figure 5.7: Model configuration 2.

The last approach is to let G_θ^1 and G_θ^2 be fully connected and thus equal through the Fourier layers while providing the split, the division of the prediction of u_h and $E\hat{u}_h$, only by the projection $Q(\cdot)$.



(a) Test and train loss using configuration 1.

(b) Test and train loss using configuration 2 and MLP connections.

Figure 5.8

Figure 5.8 (b) shows the lowest loss obtained using the connective structure and considering three different ways of connection: sum, activation and MLP. The model and optimization configuration correspond to the best obtained in experiment 1. The difference in 5.8 (a) and (b) clearly illustrates that sharing parameters θ in the prediction of the two image targets helps the model learn the generalized features of the data. We also note that the connective structure reduces the test error with an additional 1% on both image predictions, compared to the one image prediction in 5.3 (d). The last alternative, providing the split only by projection, is not reported since the loss was much higher then that of model configuration 1 and 2. A characteristic feature of the training curves provided is that the test loss does not increase but remains approximately constant while the training loss continues to decrease. Since this indeed is the case even though we drastically increase the number of learnable parameters, we suspect that the small learning rate in addition to the filtering procedure of the Fourier layer leads to the model overfitting on the filter rather than the data. The reasoning thus implies that the weights are not capable of simply constructing the target, but can only change the filter to which best predict

the target. This intuition and the shape of the test curves match that in the literature (See e.g. Li et al., 2020b, page. 9) and (Figure 3. Li et al., 2020a).

5.3 Experiment 3: Increasing data complexity

Having a model which is able to predict the features of the map $G(\Lambda) \mapsto (u, \hat{u})$ with good precision averaging at 3% relative error we want to see if the model is able to generalize for a higher number of terminal segments. We increase the number of terminal segments such that the data set consist of 10000 samples with a uniform distribution of the number of terminal segments between 2 and 10. Increasing the resolution to 128x128 we ensure that all features of the data are presented to the model. We train a model using the configuration which obtained the best predictions in Experiment 2. and obtain a mean relative error of 5%. A question then arises of how the increase in error is related to the increase in the number of terminal segments. To measure this relation we evaluate the trained model on input corresponding to different numbers of terminal segments. Since the data used to train the model is

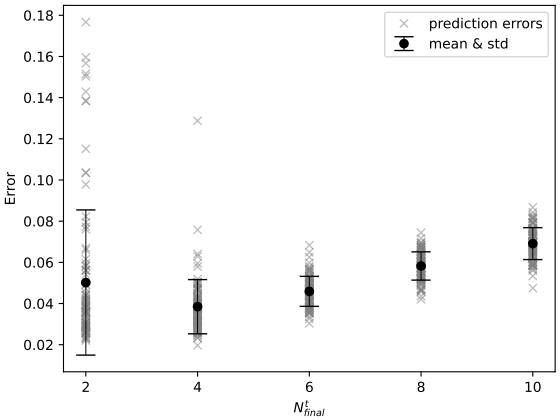


Figure 5.9: Relative error for the different number of terminal segments.

not sorted for the number of terminal segments we generate new data. We let $N^t_{final} \in [2, 4, 6, 8, 10]$ and generate 100 samples at resolution 64×64 for each value of terminal segments. Figure 5.9 illustrate how the relative error changes when the pre-trained model predicts on samples with different numbers of terminal segments. We notice two features in the plot, the error tends to increase when the number of terminal segments increases and the range of the error decreases significantly before it stabilises. Since the model is trained on a uniform distribution of the number of terminal segments it should not "prefer" one over the other. Since there is a relationship of the output from a few terminal segments to a high number of terminal segments, it is reasonable to presume that the model, to some extent contains trained parameters which generalize independent of the number of terminal segments. In this scenario, the model would predict best on the average number of terminal segments, simply because it is the mean of the sample distributions in the training set. Recalling

the three worst predictions in Figure 5.5 we suspected that for a few terminal segments occurrences of outliers with less apparent bifurcation made it difficult for the model to separate the root and terminal vessels. Investigating the input corresponding to the outliers in Figure 5.9 we see that this is the case. Input where the bifurcation is less apparent and one terminal segment lies close to the edge and hence has a similar appearance as the root leads to higher error.

5.4 Experiment 4: Discretization independence and execution time

As previously mentioned in chapter 4 a key feature of the Fourier neural operator is that it is resolution independent. We can train on one resolution and predict on another. To illustrate this capability we create 10 samples with high refinement $h = 0.001$, $N_{final}^t = 3$ and interpolate to $N_x \times N_y = 512 \times 512$. Different resolutions is down-sampled from this set and we use the best model obtained in Experiment 2, which is trained on resolution 64×64 to measure the relative error between resolutions.

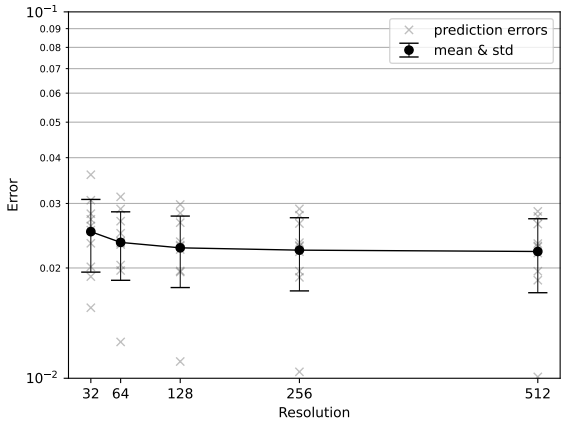
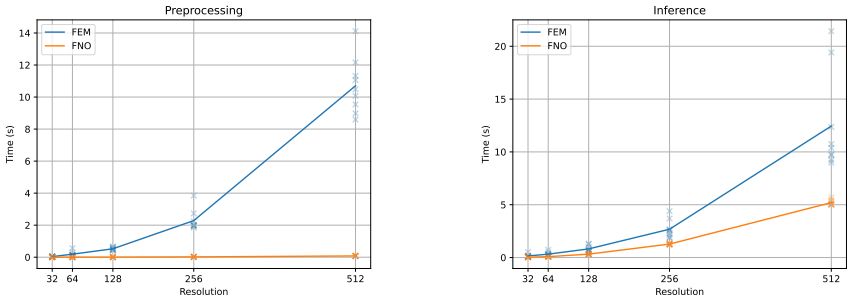


Figure 5.10: Relative error in logarithmic scale for different resolutions.

Figure 5.10 shows that the model is able to generalize to higher resolution and furthermore that the error is consistent for the different resolutions. This is a great advantage for the practical use of the model, unlike traditional solvers like the finite element solver we used to generate the data the model is not restricted to a certain discretization of the input and can thus be reused for prediction on different resolutions. This also offers an advantage compared to using other neural networks, like a U-net structure of convolutional neural networks.
Based on these results, the model G_θ can offer time-sparing advantages compared to the traditional finite element solver. Indeed, recalling the flow chart Figure 4.2 we can replace the partitioning of the geometry $(\mathcal{P}_h, \mathcal{I}_h)$ and the finite element solver by producing the distance representation of Λ and feed it directly into the model G_θ.



(a) Computing time vs resolution in preprocessing.

(b) Computing time vs resolution in inference.

Figure 5.11: Time differences in the finite element method and the proposed Fourier neural operator architecture, measured on CPU.

Resolution	32	64	128	256	512
square elements in $(N_x \times N_y)$	1024	4096	16384	65536	262144
triangle elements in $(\mathcal{P}_h, \mathcal{I}_h)$	1289	4627	17943	69637	273633

Table 5.3: Number of elements in the unit square Ω.

Figure 5.11(a) compares the time difference in preprocessing i.e. in creating the partitioning $(\mathcal{P}_h, \mathcal{I}_h)$ and in obtaining the distance representation of a simple geometry Λ. Figure 5.11(b) shows the time difference in inference, i.e. in obtaining the solution $(u_h, \hat{u}_h)$ from the finite element solver and the partitioning $(\mathcal{P}_h, \mathcal{I}_h)$ and in obtaining the solution $Y = G_\theta(X)$ from the distance representation D_Λ. We use the architecture corresponding to the best predictions in Experiment 2 and to make $(\mathcal{P}_h, \mathcal{I}_h)$ comparable to the square resolution $N_x \times N_y, N_x = N_y$ we use a simple Y-shaped structure on Λ, and let the refinement parameter be given as $h \approx 1.5/N_x = 1.5/N_y$. Since the Gmsh software refits the mesh ensuring $h_{max} < 1.5/N_x$ while preferring equilateral triangle elements the resolutions are not directly comparable, Table 5.3 contains the number of elements, triangle and square for the different discretizations. Due to inherit randomness in measuring executing time on a CPU we measure over 10 consecutive runs, and the highlighted curves in the plot are the mean. Compilation time is excluded. We clearly see that the computational expense in preprocessing increases more rapidly for the finite element model as a function of refinement than the neural network model does as a function of resolution. The difference in inference is less prominent but we do see a slight trend benefitting the machine learning model and we suspect that the differences in inference become more apparent for even higher refinement.

5.5 Conclusion and future work

Our objective was to investigate if a neural network, and specifically if the Fourier neural operator could learn the relationship between vascular geometry and fluid interaction in the tissue.

With the method of CCO, we were able to generate realistic vascular trees,

and with a simplified 2D-1D representation of the tissue and vasculature, we derived a mathematical model describing the interaction of fluid through diffusion and exchange. Fixing the physically motivated boundary conditions and only varying the geometry of the vasculature we produced sets of data consisting of input and target samples. The input was obtained by the distance representations of the geometric position of the vascular trees in the confining unit square. The target was the pressures obtained by solving the derived mathematical model of fluid interaction by an investigated finite element solver.

We then trained and tested the Fourier neural operator through a series of experiments. First, we used a data set consisting of 2-3 terminal vessels with one-image targets. We achieved high accuracy (mean relative error of 4%, measured over 2500 test samples). Next, we investigated different configurations of the FNO that allowed for two-image output and the best-performing configuration was able to further reduce the error by 1%. With a trained model at hand, we saw agreement with the expected discretization independence of the model and saw that the model could generalize to a higher number of vessels in the vascular trees. Lastly, we demonstrated the time-sparing advantage a pre-trained Fourier neural operator can offer in preprocessing and inference compared to the conforming finite element method used to generate the data.

Inference

The presented machine learning model offers a different way in both preprocessing and inference to obtain the pressure distributions compared to finite element solvers. Since we only measured the time difference in inference for our specific finite element solver it would be interesting to compare the inference time to the Fourier neural operator and alternative finite element solvers provided in (Koch et al., 2020) and (Gjerde et al., 2020).

From 2D to 3D

Throughout this thesis, we have restricted ourselves to the two-dimensional setting, but every step in the previous chapters is applicable in three dimensions and indeed this is where the potential realization of the physical application lies. The extension of the method of CCO to 3D is provided in (Karch et al., 1999). Since the method of CCO is based on the modelling of straight cylindrical tubes following Poiseuille's Law it can also be argued that 3D is the more natural setting in which to apply the method, changing only the cost to the actual volume and the spatial coordinates. The derivation of the traditional approach to obtain the fluid interactions was simplified to two dimensions and thus the methodology used in the derivation applies to 3D, specifically (Cerroni et al., 2019; Masri et al., 2023). The Fourier neural operator can be used on 2D cross sections and by putting them together we can obtain a 3D map of the pressures. A natural step in future work is thus to extend to a three-dimensional setting.

Including time

We have only considered stationary maps of the pressure but both the fluid equations and the Fourier neural operator can be extended with a temporal

variable. Interestingly, an experiment in (Li et al., 2020a) shows that the FNO is not only discretization invariant in the spatial domain but also the temporal domain when considering the Navier-Stokes equations for a viscous, incompressible fluid in vorticity form on the unit torus.

Increasing the complexity of the fluid equations

In addition to the two previous subsections, there are numerous ways to increase the complexity of the boundary value problem in the pursuit of obtaining more realistic descriptions of the fluid interaction. We do not believe that a higher dimensional representation of the vasculature is the way forward because of the computational cost of deriving the discretization. We rather believe that the radii could be represented as a parameter in the coupling factor, i.e. $\gamma = \gamma(r)$ which fits the mathematical framework presented in section 3.2. Furthermore, we have only looked at purely diffusion-driven flow but (Masri et al., 2023) suggests that fluid, specifically in brain vasculature is also driven by convection. This paper also includes time and another dimension to describe the transport of solutes in a 3D-1D-1D model.

Inverse problem

Presume we have a trained Fourier neural operator, possibly trained on vasculature obtained by the method of CCO. We denote this model G_θ and for simplicity of notation let $G_\theta(\mathbf{x}) : D_\Lambda \to U$, where D_Λ as previously denotes the distance representation of Λ and U is a pressure distribution. Additionally, we let N_θ denote a parametric map $N_\theta : \Omega_j \to B_\Lambda$ where Ω_j is the discretization of some domain Ω and B_Λ is a binary representation of Λ. Now, considering a scenario where we know U but not Λ, we can then aim to minimize

$$\min_{\theta \in \Theta} C[G_\theta(D_\Lambda(N_\theta))(\mathbf{x}), U(\mathbf{x})]$$

for some choice of cost function C to obtain the geometry Λ that best fits the pressure distribution U. Previously, we have only considered a scenario where we want to use the method of CCO to fill in the missing information of vasculature extracted through medical images but this approach possibly provides another way of filling in the missing information of the geometry. By using direct measurements of partial pressure as obtained in (Mächler et al., 2022) and by having a fast differentiable model G_θ we can find the geometry that best fits the measurements.

Real microvasculature

Lastly, we illustrate a microvascular system by providing an image of a digitized network of 15.000 vessels obtained from one cubic millimetre of the brain cortex of mice.

The microcirculatory network of vessels in Figure 5.12 illustrates the sheer complexity of these networks and thus motivates the use of different scientific disciplines, *in vivo*, *in vitro* and *in silico* to be able to get one step closer in understanding the distribution of blood in these efficient, highly integrated systems.

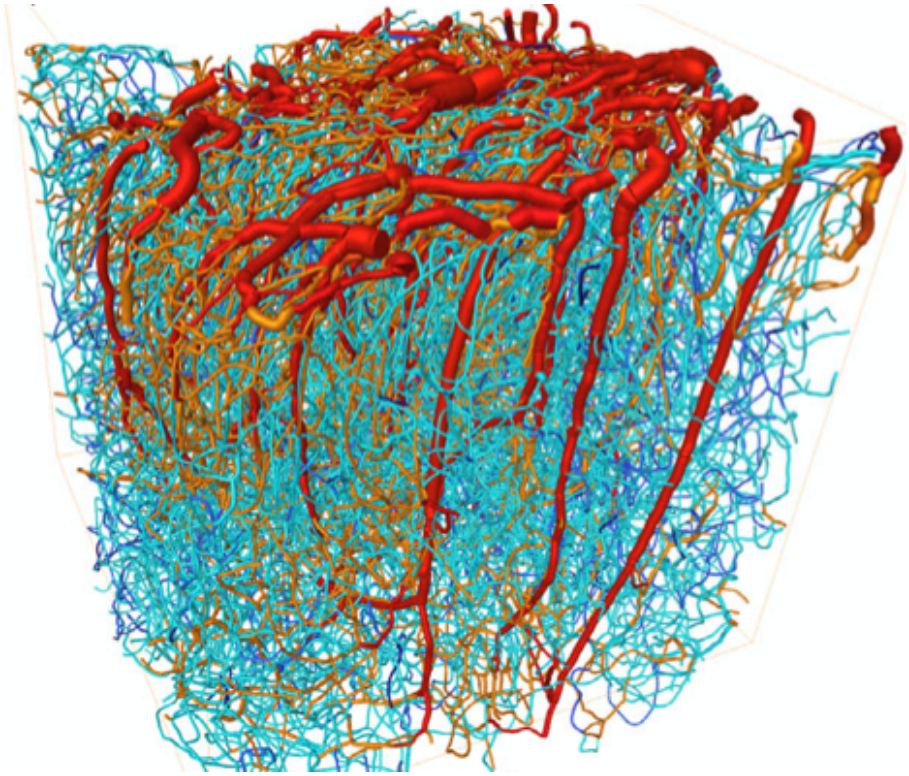


Figure 5.12: $1mm^3$ microvascular network of the mouse cortex. Image obtained from (Goirand et al., 2021). The colouring represents the simulated blood flow rate, red has a high flow rate and dark blue has a low flow rate.

Appendices

APPENDIX A

Numerical experiments of the harmonic exstension

Implementation of the harmonic extension

To ensure the correct implementation of the harmonic extension we measure the difference

$$\text{diff} = \|E\hat{u}_h - \hat{u}_h\|_{L^2(\Lambda)} \tag{A.1}$$

between the solution Eu_h and $\hat{u}_h$ on a random 1D geometry Λ with 37 terminal segments. Since $\hat{u}_h$ is enforced through Dirichlet on the boundary, we know that the difference should be machine precision when computing the solutions using P_1 elements. This is because every degree of freedom on Λ is set by the Dirichlet condition, while the difference using P_2 elements should converge for higher mesh refinement. We also measure the relative error (3.92) with the same mesh refinements given in figure (3.15). Figure (A.1) shows that we obtain the

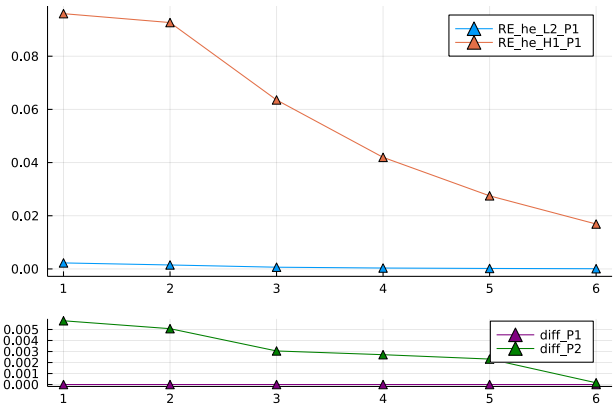


Figure A.1: Relative error (3.92) and difference (A.1) over mesh refinements.

expected results when evaluating the implementation of the harmonic extension.

Convergence of u_h to the harmonic extension

Recalling our energy functional

$$E_f(u, \hat{u}) = \frac{1}{2}\int_\Omega (\nabla u)^2 \, dx + \frac{1}{2}\int_\Lambda (\nabla \hat{u})^2 \, ds + \frac{\gamma}{2}\int_\Lambda (u - \hat{u})^2 \, ds \tag{A.2}$$

and seeking the minimizer

$$G : \min_{\vec{u} \, \in \, \mathcal{U}} E_f(\vec{u}; \Lambda), \tag{A.3}$$

it is natural to suspect that $u|_\Lambda - \hat{u} \to 0$ as $\gamma > 0$ increases since the last term in (A.2) obtains its minimum when $u|_\Lambda = \hat{u}$. Hence, the last expression can be thought of as weak enforcement of $u|_\Lambda = \hat{u}$ and the solution u_h converges to $E\hat{u}_h$. Numerically measuring the difference between $E\hat{u}_h$ and the solution u_h we should obtain a decreasing difference for increasing coupling factor γ.

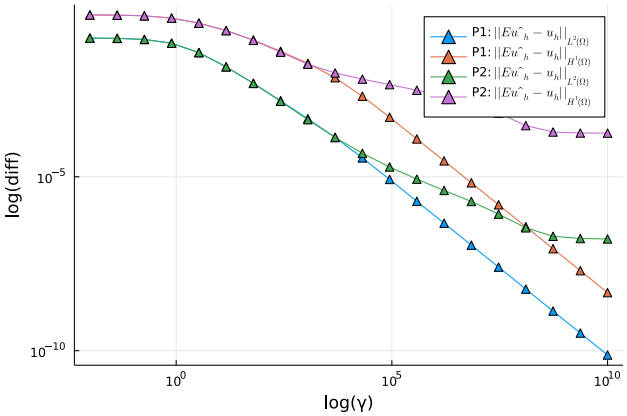


Figure A.2: log-log plot of difference $\|Eu_h - u_h\|_\Omega$ as a function of coupling factor γ, measured in the H^1 and L^2 norm, using P1 and P2 type elements.

Figure A.2 Shows the expected behavior of the difference $\|Eu_h - u_h\|$. For P_1 elements the difference goes to zero, while for P_2 elements the difference converges to approximatly 10^{-5} because there is some approximation error assosiated with the enforcement of $\hat{u}_h$ on Λ.

APPENDIX B

Rapidly exploring random trees

We present an alternative to the method of CCO to obtain the different geometries Λ. The method is faster but less realistic as it is a stochastic procedure and does not follow any physical laws as the method of CCO.
Inspired by the Rapidly Exploring Random Tree (RRT) algorithm (LaValle, 1998) we have constructed a way of drawing points in the plane such that it exhibits the branching features of vasculature.

Algorithm 4 RRT

$\mathbf{p}_1 = (x_1, y_1)$ ▷ Starting point
  i = 2
  while $i \leq N$ **do**
    Draw random point $\mathbf{p}_d$ following uniform distribution $U(0,1)$ in each coordinate
    $n = \text{argmin}_{j \in \{1,..,i-1\}} \mathbf{a}_j - \mathbf{b}$ ▷ Get closest point $\mathbf{a}_j$ to $\mathbf{p}_d$
    $\theta = \arctan^{-1}(\mathbf{a}_n, \mathbf{p}_d)$
    $\mathbf{p}_i = \mathbf{p}_n + (\cos(\theta), \sin(\theta))$ ▷ New point
    $i = i + 1$
  end while

Note that the new point is always one unit length from the previous point, while the randomness is in the angle. By changing the spread of the random numbers b we can vary the stochastic nature of the points. Algorithm 5 shows how we

Algorithm 5 Connect RRT

  for $i \in \{2, .., N\}$ **do**
    iter = 0
    for $j \in \{1, .., N\}$ **do**
      if $i > j$ & $|\mathbf{p}_i - \mathbf{p}_j| < d$ **then**
        iter = iter +1
        if iter < 2 **then**
          connections $\leftarrow (j, i)$ ▷ Store index as low to high
        end if
      end if
    end for
  end for

connect the points, looking at a specific point $\mathbf{p}_i$ we allow for one connection with a lower indexed point $\mathbf{p}_j$ within a given distance d. From Algorithm 4 we see that each new point is 1 unit away from its closest point and therefore we simply set $d = 1+\text{tol}$. Algorithm 4 and Algorithm 5 provides a specified way to generate different 1D geometries. The resulting geometries exhibit four desired features *continuity*, *randomness*, *none loops* and *none intersections*
We enforce *none loops*, by only allowing two connections per point. With the stochastic nature of our 1D geometry generation, it sometimes happens that the connecting lines from algorithm 2. intersect, although this is an anomaly that can happen in real vasculature, meshing such domains require special attention. We, therefore, add a tolerance on how close the points $\{\mathbf{p}\}_i^N$ can be to each other - making the intersection less likely to happen and additionally having the upside of producing more equilateral triangle mesh elements. Additionally, we include a safety measure of neglecting seeds producing intersecting lines.

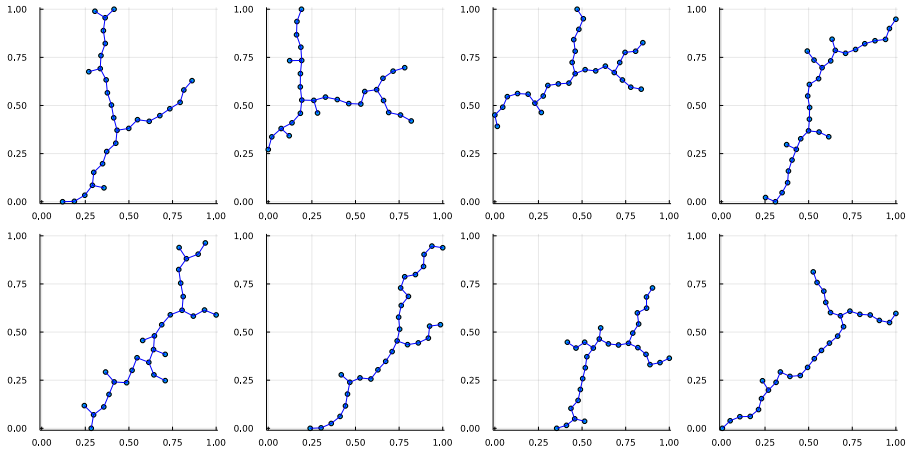


Figure B.1: Eight random 1D geometries generated with algorithm 1 and 2. and normalized s.t. $x, y \in [0, 1]$. $N = 30$, $(x_1, y_1) = [U(0, N/4), 0]$ or $[0, U(0, N/4)]$.

Figure B.1 shows eight different geometries obtained by Algorithm 4 and Algorithm 5. The connective routine provides only bifurcations similar to the method of CCO, but the number of bifurcations are in this method arbitrary.

Bibliography

Badia, S., & Verdugo, F. (2020). Gridap: An extensible finite element toolbox in julia. *Journal of Open Source Software*, vol. 5no. 52, 2520.

Barnsley, M. F., & Demko, S. (1985). Iterated function systems and the global construction of fractals. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, vol. 399no. 1817, 243–275.

Bassingthwaighte, J. B., King, R. B., & Roger, S. A. (1989). Fractal nature of regional myocardial blood flow heterogeneity. *Circulation Research*, vol. 65, 578–590.

Braess, D. (2007). *Finite elements: Theory, fast solvers, and applications in solid mechanics* (3rd ed.). Cambridge University Press.

Brown, J., West, G., Santa Fe Institute (Santa Fe, N., & on "Scaling in biology : from organisms to ecosystems", S. (2000). *Scaling in biology*. Oxford University Press.

Cattaneo, L., & Zunino, P. (2014). A computational model of drug delivery through microcirculation to compare different tumor treatments. *International journal for numerical methods in biomedical engineering*, vol. 30no. 11, 1347–1371.

Cerroni, D., Laurino, F., & Zunino, P. (2019). Mathematical analysis, finite element approximation and numerical solvers for the interaction of 3d reservoirs with 1d wells. *GEM-International Journal on Geomathematics*, vol. 10, 1–27.

Cooley, J. W., & Tukey, J. W. (1965). An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, vol. 19, 297–301.

D'Angelo, C. (2007). *Multiscale modelling of metabolism and transport phenomena in living tissues* (tech. rep.). EPFL.

D'Angelo, C. (2012). Finite element approximation of elliptic problems with dirac measure terms in weighted spaces: Applications to one-and three-dimensional coupled problems. *SIAM Journal on Numerical Analysis*, vol. 50no. 1, 194–215.

Elman, H. C., Silvester, D. J., & Wathen, A. J. (2014). *Finite elements and fast iterative solvers: With applications in incompressible fluid dynamics*. Numerical Mathematics; Scie.

Evans, L. C. (1998). *Partial differential equations*. American Mathematical Soc., 2010.

Geuzaine, C., & Remacle, J.-F. (2009). *Gmsh: A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities* (Version 4.10.5).

Gjerde, I. G., Kumar, K., & Nordbotten, J. M. (2020). A singularity removal method for coupled 1d–3d flow models. *Computational Geosciences*, vol. 24, 443–457.

Goirand, F., Le Borgne, T., & Lorthois, S. (2021). Network-driven anomalous transport is a fundamental component of brain microvascular dysfunction. *Nature communications*, vol. 12no. 1, 7295.

Gottlieb, M. (1990). Modelling blood vessels: A deterministic method with fractal structure based on physiological rules. *[1990] Proceedings of the Twelfth Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 1386–1387.

Grisvard, P. (2011). *Elliptic problems in nonsmooth domains*. SIAM.

Guo, X., Li, W., & Iorio, F. (2016). Convolutional neural networks for steady flow approximation. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 481–490.

Jaquet, C., Najman, L., Talbot, H., Grady, L., Schaap, M., Spain, B., Kim, H. J., Vignon-Clementel, I., & Taylor, C. A. (2018). Generation of patient-specific cardiac vascular networks: A hybrid image-based and synthetic geometric model. *IEEE Transactions on Biomedical Engineering*, vol. 66no. 4, 946–955.

Karch, R., Neumann, F., Neumann, M., & Schreiner, W. (1999). A three-dimensional model for arterial tree representation, generated by constrained constructive optimization. *Computers in biology and medicine*, vol. 29no. 1, 19–38.

Karch, R., Neumann, F., Neumann, M., & Schreiner, W. (2000). Staged growth of optimized arterial model trees. *Annals of biomedical engineering*, vol. 28, 495–511.

Karniadakis, G. E., Beskok, A., & Gad-el-Hak, M. (2002). Micro flows: Fundamentals and simulation. *Appl. Mech. Rev.*, vol. 55no. 4, B76–B76.

Kingma, D. P., & Ba, J. (2017). Adam: A method for stochastic optimization.

Koch, T., Schneider, M., Helmig, R., & Jenny, P. (2020). Modeling tissue perfusion in terms of 1d-3d embedded mixed-dimension coupled problems with distributed sources. *Journal of Computational Physics*, vol. 410, 109370.

Kodama, Y., Aoki, H., Yamagata, Y., & Tsubota, K. (2019). In vitro analysis of blood flow in a microvascular network with realistic geometry. *Journal of Biomechanics*, vol. 88, 88–94.

Kovachki, N., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2021). Neural operator: Learning maps between function spaces. *arXiv preprint arXiv:2108.08481*.

Kuchta, M., Nordaas, M., Verschaeve, J. C. G., Mortensen, M., & Mardal, K.-A. (2016). Preconditioners for saddle point systems with trace constraints coupling 2d and 1d domains. *SIAM Journal on Scientific Computing*, vol. 38no. 6, B962–B987.

Laurino, F., & Zunino, P. (2019). Derivation and analysis of coupled pdes on manifolds with high dimensionality gap arising from topological model reduction. *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 53no. 6, 2047–2080.

LaValle, S. M. (1998). Rapidly-exploring random trees : A new tool for path planning. *The annual research report*.

LeCun, Y., Bottou, L., Orr, G. B., & Müller, K.-R. (2002). Efficient backprop. In *Neural networks: Tricks of the trade* (pp. 9–50). Springer.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020a). Fourier neural operator for parametric partial differential equations.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020b). Neural operator: Graph kernel network for partial differential equations.

Linninger, A., Gould, I., Marinnan, T., Hsu, C.-Y., Chojecki, M., & Alaraj, A. (2013). Cerebral microcirculation and oxygen tension in the human secondary cortex. *Annals of biomedical engineering*, vol. 41, 2264–2284.

Loshchilov, I., & Hutter, F. (2017). Fixing weight decay regularization in adam.

Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021). Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, vol. 3no. 3, 218–229.

Maso Talou, G., Safaei, S., Hunter, P., & Blanco, P. (2021). Adaptive constrained constructive optimisation for complex vascularisation processes. *Scientific Reports*, vol. 11.

Masri, R., Zeinhofer, M., Kuchta, M., & Rognes, M. E. (2023). The modelling error in multi-dimensional time-dependent solute transport models.

Meyer, E. P., Ulmann-Schuler, A., Staufenbiel, M., & Krucker, T. (2008). Altered morphology and 3d architecture of brain vasculature in a mouse model for alzheimer's disease. *Proceedings of the national academy of sciences*, vol. 105no. 9, 3587–3592.

Murray, A., & Dinsdale, G. (2016). Imaging the microcirculation. *Microcirculation*, vol. 23no. 5, 335–336.

Murray, C. D. (1926). The physiological principle of minimum work applied to the angle of branching of arteries. *The Journal of general physiology*, vol. 9no. 6, 835–841.

Mächler, P., Fomin-Thunemann, N., Thunemann, M., Sætra, M. J., Desjardins, M., Kılıç, K., Amra, L. N., Martin, E. A., Chen, I. A., Şencan-Eğilmez, I., et al. (2022). Baseline oxygen consumption decreases with cortical depth. *PLoS Biology*, vol. 20no. 10, e3001440.

Nekka, F., Kyriacos, S., Kerrigan, C., & Cartilier, L. (1996). A model of growing vascular structures. *Bulletin of Mathematical Biology*, vol. 58no. 3, 409–424.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library.

Peyrounette, M., Davit, Y., Quintard, M., & Lorthois, S. (2018). Multiscale modelling of blood flow in cerebral microcirculation: Details at capillary scale control accuracy at the level of the cortex. *PloS one*, vol. 13no. 1, e0189474.

Quarteroni, A., Sacco, R., & Saleri, F. (2006). *Numerical mathematics*. Springer Berlin Heidelberg.

Renardy, M., & Rogers, R. C. (1993). An introduction to partial differential equations.

Salsa, S. (2016). *Partial differential equations in action: From modelling to theory*. Springer International Publishing.

Schreiner, W., & Buxbaum, P. (1993). Computer-optimization of vascular trees. *IEEE Transactions on Biomedical Engineering*, vol. 40no. 5, 482–491.

Sherman, T. F. (1981). On connecting large vessels to small. the meaning of murray's law. *The Journal of General Physiology*, vol. 78, 431–453.

Sutera, S. P., & Skalak, R. (1993). The history of poiseuille's law. *Annual review of fluid mechanics*, vol. 25no. 1, 1–20.

van Neerven, J. (2022). *Functional analysis*. Cambridge University Press.

Viquerat, J., & Hachem, E. (2020). A supervised neural network for drag prediction of arbitrary 2d shapes in laminar flows at low reynolds number. *Computers & Fluids*, vol. 210, 104645.

West, B. J., & Goldberger, A. L. (1987). Physiology in fractal dimensions. *American Scientist*, vol. 75no. 4, 354–365.

Whitaker, S. (1986). Flow in porous media i: A theoretical derivation of darcy's law. *Transport in porous media*, vol. 1, 3–25.